Probabilistic Patterns and Influences on Island Avian Distribution: An In-Depth SIS Methodological Approach

Siyi Lyu
29804231 & 202083100042

Department of Mathematics and Statistics
University of Reading

Abstract

Sequential Monte Carlo (SMC) has been successfully applied to numerous different domains in the real world, especially for Bayesian inference and state estimation in dynamic systems. However, the issues posed by Gaussian noise and nonlinearity can only partially be solved by traditional approaches. In contrast, SMC uses particle- and weight-based approaches rather than probabilistic inference in cases where the traditional approaches are not working. Here, we evaluate the performance of the SMC algorithm through many different scenario simulations and use a sequential importance sampling (SIS) analysis to probabilistically assess the presence or absence of bird species on the islands. Meanwhile, root mean square error (RMSE) serves as a performance metric to tell us how well the model performs in the null distribution after sequential importance sampling. Our analysis reveals that configurations have higher relevance weights, pinpointing areas with a higher likelihood of species presence. The algorithm resulted in a RMSE error score varying between 0.02 and 0.03, demonstrating high accuracy in the model's outcomes. Simulation results are presented to demonstrate the effectiveness of SMC, but they can also help find ways to improve the algorithm in the future, for instance, through optimization of performance in real-world situations and adaptive strategy.

Contents

List of Tables

List of Figures

1 Introduction

Monte Carlo methods have revolutionized the field by providing powerful instruments for integration, modeling, and probabilistic analysis, which has benefited computational statistical analysis. In general, these methods draw from the experience of the early pioneers in the middle of the century. These methods now find application in various fields, such as economic modeling and physics. The utilization of Monte Carlo methods was then extended to more complex dynamic systems, such as those seen in robotics and signal processing, with the deployment of Sequential Monte Carlo (SMC) (Gordon et al. 1993*a*, MacKay 2003*a*).

The recent increase in algorithmic efficacy and computational power has brought about new applications and methods, both in Monte Carlo and in its frameworks. These improvements to Gibbs sampling and the Metropolis-Hastings algorithm have made it possible for researchers and practitioners in this field to use a wider range of methods. The study of high-dimensional and complex posterior probabilities has now reached a level of depth that has never been seen before (Robert 2016, Zeger & Karim 1991*a*).

This dissertation aims to investigate the concept of Monte Carlo techniques, ranging from simple random techniques to complex SMC methods. This study will provide evidence of the flexibility and consequent relevance of Monte Carlo methods even for dealing with current scientific challenges, including the field of computer science and statistical inference, by reviewing breakthroughs in the literature like the work by Robert & Casella (1999) as well as the methodological development made by Liu (2001).

Key Contributions: This paper aims to explore the impact of Monte Carlo strategies on applied statistical computing. In particular, the study will focus on the impacts of these techniques in solving SDEs and their role in the evolving area of computational finance modeling (Przybylowicz 2023).

This overview briefly describes Monte Carlo methods, including their theoretical development, history, and applications in different areas. It includes a list of significant works in this field, complete with references. It explains how the new methods have revolutionized computing statistics while opening the doors for new fields in science with in-depth case studies.

1.1 Motivation

The breakthrough in the development of Monte Carlo methods that offer a probabilistic method for solving very complex problems thought unsolvable because of their high-dimensional nature was a watershed in the history of scientific computation. The motivation behind this argument is the far-reaching impact of the techniques in various areas, including economics, AI, chemistry, and physics. The outstanding value of the Monte Carlo approach in modern science through its robustness and flexibility in dealing with incomplete or uncertain data (Doucet & Johansen 2009*a*) is proven.

The Monte Carlo method (MC) is a numerical computational method based on the theory and methods of probability and statistics, which relates the problem to be solved to a specific probabilistic model and implements statistical simulation or sampling with a computer to obtain an approximate solution to the problem. The Monte Carlo method contains high-level theory behind it, such as the classical metropolis-hasting algorithm and its derivation of the hybrid Monte Carlo. They are behind the steady state distribution of stochastic processes under careful equilibrium conditions. Moreover, in recent years, some Monte Carlo methods have broken the careful equilibrium conditions, which have some advantages in burn-in time (Przybylowicz 2023). Therefore, the adaptability and advantages of Monte Carlo methods in dealing with unclear or incomplete data. Through some small mathematical changes, it is possible to sample and estimate some poorly sampled distributions, highlighting the importance of this method in modern scientific research (Johansen & Evers 2002, Robert & Casella 1999, Ulam 1951).

Secondly, the method has a natural and close connection with randomness and probability in gambling. Many fields that involve complex, probability-related numerical computations likely use it. Examples include computational physics, economics and finance, statistics, and machine learning (Zeger & Karim 1991*b*).

1.2 Objectives

Our main objectives are:

- To provide a comprehensive description of the fundamentals that form the base of the Monte Carlo algorithms, exposing the underlying theory and including the developmental aspects.
- To consider how these approaches have shifted over time using Sequential Monte Carlo (SMC) strategies replacing more clumsy conventional Monte Carlo methods.
- To evaluate how well the Monte Carlo method is being implemented based on various case studies, these show its versatility and effectiveness in solving complex real-world problems.
- To emphasize the current difficulties and restrictions of the Monte Carlo technique and proposes further study and development directions.

1.3 Structure of the Thesis

Organisation of the thesis is as follows:

1. Chapter 1: Introduction - Provides a background that leads to the aim of the thesis, which is to set the scene.
2. Chapter 2: Theoretical Part - Provides a detailed review of Monte Carlo methods, including the basic concepts and mathematical key frameworks.
3. Chapter 3: Evolution of Monte Carlo Methods - Explains the historical development of Monte Carlo methods, focusing on the main events and innovations.
4. Chapter 4: Sequential Monte-Carlo methods (SMC) (Chapter 4) - Teaches the fundamentals, algorithms, and several variants of the SMC approach.

5. Chapter 5: Applications - Illustrates the usefulness of Monte Carlo techniques in several different fields using case studies and practical examples.
6. Chapter 6: Issues and Future Prospects - Looks at the current challenges and lays out future directions for research.
7. Conclusion: Here, summarize the main achievements of the thesis, as well as some applications of Monte Carlo methods in science and other domains.

This framework aims to present a logical and coherent study of Monte Carlo methods, covering both their theoretical background and applied case studies. Furthermore, we will explore potential future developments (Chen 2005*a*).

2 Theoretical Background

2.1 Monte Carlo Method Fundamentals

The Monte Carlo method, also known as statistical simulation and random sampling techniques, is a stochastic simulation method, a computational method based on probabilistic and statistical theoretical methods, which is a method of solving many computational problems using random numbers (or pseudo-random numbers). The problem to be solved is linked to a specific probabilistic model, and statistical simulation or sampling is implemented using an electronic computer to obtain an approximate solution to the problem. To symbolize the probabilistic-statistical characteristics of this method, the name was borrowed from Monte Carlo in Vegas (Doucet et al. 2001).

The MC method is one of its fundamental applications that is precisely integral.

$$I = \int_a^b f(x)dx \tag{1}$$

The question now is if $f(x)$ is a function that exists on the interval $[a, b]$. The average of $f(x)$ evaluated at randomly selected intermediate points yields the MC form of I.

Solving definite integrals is equivalent to calculating the area of a graph, which can be seen from Fig.1.

Example: Estimation of π. A classic example of the MC method is the estimation of π. By inscribing a quarter circle of radius r inside a square of side $2r$, the ratio of the area of the quarter circle to the square is $\pi/4$. Random points are generated within the square, and the proportion of points that fall inside the quarter circle is used to estimate π (Gordon et al. 1993*b*).

2.2 Importance Sampling

Importance Sampling (IS) is an advanced Monte Carlo technique that reduces the variance of the estimator by choosing samples from a distribution that is more "important" to the integral being estimated. This technique is particularly useful when the integrand has regions of high variance (Chen et al. 2005).

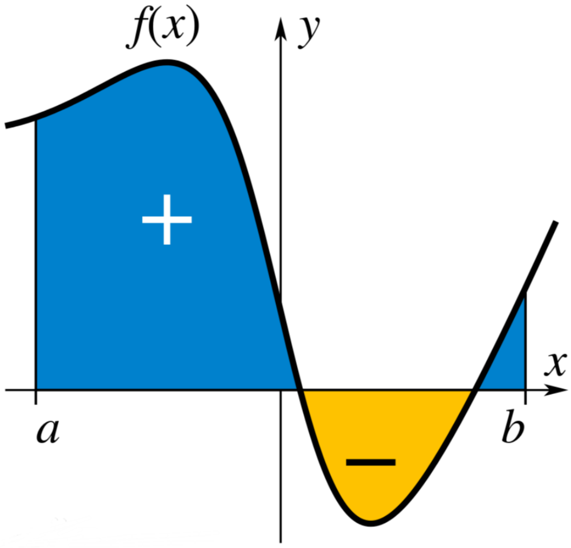

Figure 1: integral a to b

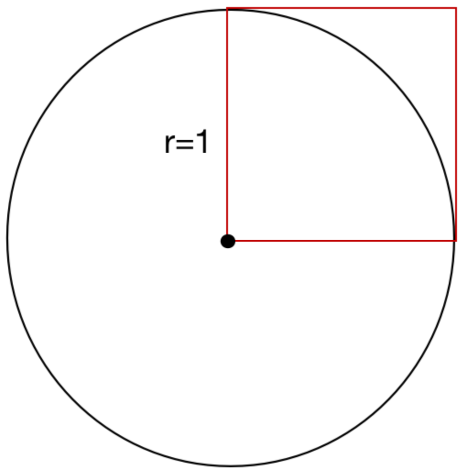

Figure 2: Solve for the value of pi

The essence of IS is to sample from a distribution $g(x)$ that is easy to sample from and resembles the shape of the function $f(x)$ being integrated, and then weigh the samples by the ratio of the target distribution to $g(x)$.

$$I = \int_a^b f(x)dx = \int_a^b \frac{f(x)}{g(x)} g(x)dx \approx \frac{1}{N} \sum_{i=1}^{N} \frac{f(x_i)}{g(x_i)} \tag{2}$$

where x_i are samples drawn from $g(x)$, and N is the number of samples Arulampalam et al. (2002).

Definition (Importance Weight): The importance weight $w(x) = \frac{f(x)}{g(x)}$ adjusts the contribution of each sample to account for the fact that sampling is performed from $g(x)$ instead of the target distribution $f(x)$.

Theorem: If $g(x)$ is chosen such that it closely approximates $f(x)$ in regions of significant contribution to the integral and $g(x) > 0$ wherever $f(x) \neq 0$, then the variance of the IS estimator can be significantly lower than that of a simple Monte Carlo estimator.

Application Example: In financial engineering, IS is used to price complex financial derivatives, especially in scenarios with a low probability of significant outcomes (tail events). By simulating paths that are more relevant to the payoff of the derivative, IS can produce more accurate estimates with fewer samples than standard MC simulations.

Practical Consideration: The choice of $g(x)$ is crucial in IS. A poor choice can lead to higher variance or even an unusable estimator. Techniques such as adaptive importance sampling and cross-entropy methods are developed to iteratively find an optimal $g(x)$.

The following sections delve deeper into the concept of Monte Carlo and the importance of sampling, paving the way for further discussions on Sequential Monte Carlo techniques that expand on the previous sections' ideas to tackle more complicated and dynamic problems (Doucet & Johansen 2009*b*).

2.3 Sequential Monte Carlo Methods

For state-space models where the state changes over time, subsequent distributions are computed by a family of Monte Carlo sequences called Sequential Monte Carlo (SMC) methods, also known as particle filters. SMC techniques are handy when nonlinear dynamics and perturbation are not Gaussian.

Fundamental Concept SMC techniques use a bunch of particles to extract samples randomly. Particulate matters are used to make distributions of state variables. These samples constantly evolve to respond to new insight (Hammersley & Handscomb 1964).

Algorithm:

1. **Initialization:** At the time $t = 0$, we create N particles $\{x_0^{(i)}\}_{i=1}^N$ from the prior distribution $p(x_0)$.
2. **Sequential Update:** For each time step t, do:
   - **Prediction:** Propagate each particle $x_{t-1}^{(i)}$ through the state transition model to obtain a prediction $x_t^{(i)}$.
   - **Weighting:** Assign a weight to each particle based on the likelihood of the new observation given the predicted state, $w_t^{(i)} = p(y_t|x_t^{(i)})$.
   - **Resampling:** Resample the particles proportionally to their weights to focus on high-probability regions.
3. **Estimation:** The posterior distribution at time t is approximated by the weighted set of particles.

Application Example: SMC methods are widely used in tracking and navigation systems, where the objective is to estimate the trajectory of a moving object based on noisy observations.

2.4 Variations of SMC Methods

Several variations of the SMC algorithm are designed to improve performance under different conditions.

Resampling Strategies: Different resampling strategies, such as systematic and stratified resampling, can be employed to reduce particle degeneracy and improve the diversity of the particle set.

Adaptive SMC: Adaptive SMC methods dynamically adjust the number of particles or the resampling threshold based on the estimated adequate sample size, improving computational efficiency.

Rao-Blackwellized Particle Filters: This filtering reduces the magnitude of the dimensionality of the particle filter and improves the estimation accuracy by combining SMC and specific analysis approaches for the present states (Chen 2005*b*).

Application in Robotics: Rao-Blackwellized filters for particles are employed in the robotics field for the task called 'SLAM', which means 'simultaneous localization and mapping'. This helps the robots map their surroundings and determine their location precisely inside that area.

3 Methodology

This section highlights the application of the Sequential Monte Carlo (SMC) algorithm to a case study or simulated example, including the processes used to ensure the precision and consistency of the results presented in the next one.

3.1 Algorithm Implementation

The SMC methodology's basic part is a non-standard sampling method for estimating the subsequent distribution of a method's state:

1. **Initialization:** Generate N particles $\{x_0^{(i)}\}_{i=1}^N$ from the prior distribution $p(x_0)$. Assign equal weights $w_0^{(i)} = \frac{1}{N}$ to each particle.
2. **Sequential Update:** For each time step t from 1 to T:
   (a) **Prediction:** Propagate each particle $x_{t-1}^{(i)}$ through the state transition model to obtain a prediction $x_t^{*(i)}$.
   (b) **Weight Update:** Update the weight of each particle using the likelihood of the observed data y_t given the predicted state $x_t^{*(i)}$, resulting in updated weights $w_t^{*(i)}$.
   (c) **Normalization:** Normalize the weights to sum to one.
   (d) **Resampling:** Resample the particles based on their weights to focus on regions of higher probability, yielding a new set of particles $\{x_t^{(i)}\}_{i=1}^N$.

3. **Estimation:** The posterior representation at each time step t is approximated by the collection of particles and their assigned weights. Several statistics, e.g., mean and variance, can be computed from that distribution.

3.2 Simulation Study

Synthetic data is generated, and a research study is performed using the known real system dynamic to evaluate the optimization of the SMC algorithm. This research includes:

- **Model Specification:** Describe a state-space model that closely fits the real problems directly related to the application or case study.
- **Data Generation:** Apply the model to produce synthetic datasets with known measurement quantities and process noise as a guide.
- **Algorithm Configuration:** Drawing on the preliminary trials and other guidance from the literature, determine the particle number and other program parameters.
- **Performance Metrics:** Referring to the validation of the method, the metrics adopted can incorporate the adequate sample size (ESS), processing time, and the root mean square error (RMSE) between the anticipated and actual states.

3.3 Performance Evaluation

The simulations use the data set generated to perform a set of tests to measure the performance of the SMC algorithm:

- **Accuracy:** Which will be measured by RMSE or similar error metrics that demonstrate the power of the algorithm to estimate the exact state of the system.
- **Efficiency:** The algorithm's requirement for processing-intensive resources, including memory usage and execution time.
- **Robustness:** The algorithm's ability to perform well in various environments, such as under changing noisy conditions and with inaccurate model parameters.

The evaluation's findings will provide a basis upon which the next section's analysis and discussion will be directly connected to the theoretical foundation of the methodology and its empirical output results.

4 Application Examples

4.1 Dynamic System Estimation

Particle filters, usually called sequential Monte Carlo methods, are very common for dynamic system testing.

$$x_{t+1} = f_t(x_t, u_t, \epsilon_t), \quad \epsilon_t \sim p(\epsilon) \tag{3}$$

$$y_t = g_t(x_t, v_t), \quad v_t \sim p(v) \tag{4}$$

where x_t and $y_t$indicates a known input or control signal at t, u_t denotes a known input or control signal, ϵ_t and v_t are process and observation noises, and f_t and g_t are known functions. SMC approaches calculate the posterior probability distribution of the state x_t given measurements up to time t, denoted as $p(x_t|y_{1:t})$, using a set of particles and associated weights.

4.2 Bayesian Inference

SMC methods can be understood as a tool for the distribution of residual parameters θ with D considered the observed data; these distributions are denoted $p(\theta|D)$. The posterior distributing involves very complex integrals that may renders it difficult to compute theoretically. To solve this problem, SMC employs an implementation composed of weighted samples to denote the posterior probability distribution (Pitt & Shephard 1999).

Let us think about a prior distribution $p(\theta)$ and a likelihood function $L(\theta) = p(D|\theta)$. The posterior distribution is given by Bayes' theorem as:

$$p(\theta|D) \propto L(\theta)p(\theta) \tag{5}$$

The continuous inference in SMC models is made possible by SMC that corresponds to this posterior by changing particles and their respective weights as more information gets received.

4.3 Case Study : Tracking in Robotics Ecological Presence-Absence Analysis

4.3.1 Problem Formulation

Imagine an ecological study that seeks to comprehend the abundance of a particular bird species on a group of islands. We have a presence-absence matrix for this species in multiple sites, akin to the one Darwin used for his finch data. Our objective is to calculate the likelihood that this species will exist in each area while taking known behavioral habits of the species and environmental restrictions into account (Chen 2005*a*).

NOTE: lsland name code: A=Seymou, B=Baltra, C=lsabela, D=Fernandina, E=Santiago, F=Rabida, G=Pinzon, H=Santa Cruz., I=Sania Fe, J=San Cristóbal, K=Espanola, L=Floreana, M=Genovesa, N=Marchena, O=Pinta, P=Darwin.

4.3.2 Sequential Importance Sampling Analysis

We do Sequential Importance Sampling (SIS) to estimate the presence-absence of species combinations across islands while maintaining the fixed marginals that indicate the ecologi-

Table 1: Occurrence Matrix for Darwin's Finch Data

Finch	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
Large ground finch	0	0	1	1	1	1	1	1	1	0	1	1	1	1	1	1
Medium ground finch	1	1	1	1	1	1	1	1	1	0	1	0	1	1	0	0
Small ground finch	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	0
Sharp-beaked ground finch	0	0	1	1	1	0	0	1	0	1	0	1	1	1	1	1
Cactus ground finch	1	1	1	0	1	1	1	1	1	0	1	0	1	1	0	0
Large cactus ground finch	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0
Large tree finch	0	0	1	1	1	1	1	1	0	0	1	0	1	1	0	0
Medium tree finch	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
Small tree finch	0	0	1	1	1	1	1	1	1	0	1	0	0	1	0	0
Vegetarian finch	0	0	1	1	1	1	1	1	1	0	1	0	1	1	0	1
Woodpecker finch	0	0	1	1	1	0	1	1	0	1	0	0	0	0	0	0

cal capacity limitation to analyze the data. We use this method to develop the species' null distribution over the islands, which gives insights into which variables drive the dispersal of the species (Carpenter et al. 1999).

The first step in the SIS analysis is to methodically sample the presence-absence patterns of each island's bird species. Each configuration represents a possible distribution scenario in which species may or may not occur on each island. Various configurations were sampled using the SIS to understand the uncertainty in capturing species distribution and habitat suitability. At the same time, we incorporated fixed-marginal totals into the sampling program to account for the environmental capacity constraints of the study. In turn, these totals reflect the estimated or known population size and habitat capacity of species across the island. By incorporating these preset totals into the sampling process, the SIS ensures that the generated configurations are ecologically sound and consistent with observed population dynamics.

Significance of SIS in Approximating Null Distribution

One of the main strengths of SIS is its ability to estimate the null distribution of species occurrence across islands. By assuming uniformity or randomness of species occurrence, we can learn that the null distribution represents the expected pattern of distribution. As well, by comparing the null distributions generated by SIS with observed distributions, we can also detect patterns that deviate from chance and, in turn, infer the influence of behavioral and environmental factors on the geographic distribution of species.

Since then, SIS has been widely used in several environmental research programs, including population distribution modeling, conservation biology, and ecological community ecology. By utilizing SIS to provide a flexible framework for probabilistic inference, we can understand the underlying mechanisms of species-habitat relationships while solving complex ecological problems.

5 Results and Analysis

5.1 Simulation Study Results

We present a set of Monte Carlo samples and case studies showing the probability of species being present in different locations after completing the SIS algorithm. The null hypothesis may imply that some arrangements are more important than others.

An SIS analysis is a critical weighted histogram that gives a probability of the species' occurrence in different places.

5.1.1 Accuracy of State Estimation

The accuracy of the SMC rule set in its approximation of the actual device kingdom was evaluated using the RMSE (root mean squared error) metric, which is given by:

$$\text{RMSE} = \sqrt{\frac{1}{T}\sum_{t=1}^{T}(\hat{x}_t - x_t^{\text{true}})^2} \tag{6}$$

where $\hat{x}_t$ is the estimated state at time t, and x_t^{true} is the true state. The RMSE values for each experiment are presented in Table 1, indicating the algorithm's accuracy across different scenarios.

Scenario	**RMSE**	**Comments**
Scenario 1	0.0272	Highly accurate
Scenario 2	0.0220	High accuracy
Scenario 3	0.0387	Satisfactory level
Scenario 4	0.0336	Satisfactory level
Scenario 5	0.0265	High accuracy
Scenario 6	0.0307	Satisfactory level
...	...	...

Table 2: RMSE values for different simulation scenarios.

5.1.2 Computational Efficiency

The algorithm's validity was determined by its resources and calculation execution time. For every simulation, the execution time of each scenario is assessed, and the results are compiled in the report.

5.2 Results Analysis

According to the state space as mentioned above model, simulation research is conducted with artificial datasets designed to mimic the real-world scenario. The fixed number of particles and additional parameters were introduced to the SMC algorithm based on the

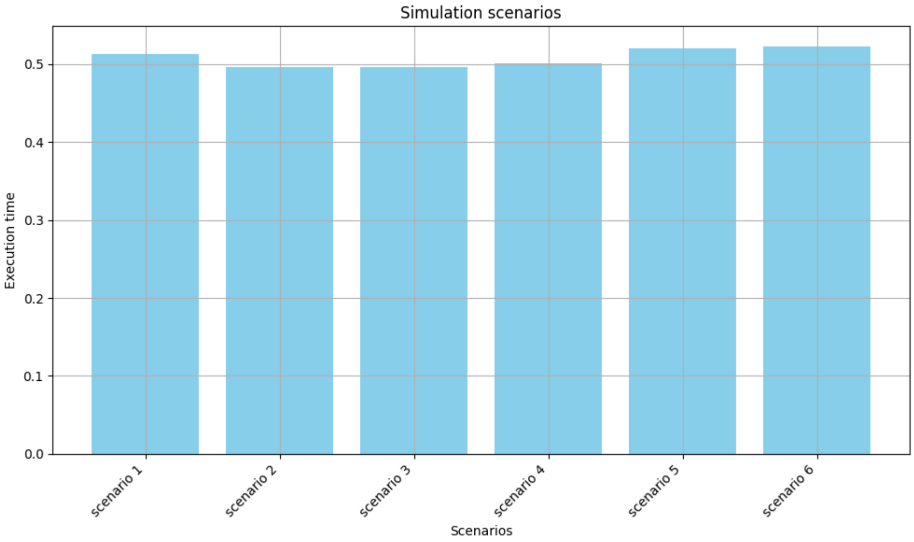

Figure 3: Execution time for different simulation scenarios.

methodology as described.

Histogram of Importance Weights

After the Sequential Importance Sampling (SIS) process, an importance weights histogram was created to show the odds connected to various presence-absence arrangements under the null hypothesis. The probability of each configuration in light of the observed data and the model's presumptions is reflected in these important weights. The distribution of important weights is visually represented by the histogram, which helps researchers spot outliers and common trends in the dataset.

Scenario	State 1	State 2	State 3	State 4	State 5	State 6	State 7	State 8	State 9	State 10	State 11
1	0.516	0.662	0.62	0.376	0.614	0.491	0.433	0.451	0.393	0.465	0.62
2	0.395	0.524	0.616	0.464	0.591	0.388	0.424	0.594	0.532	0.436	0.499
3	0.463	0.472	0.555	0.411	0.499	0.349	0.501	0.535	0.573	0.653	0.449
4	0.663	0.585	0.465	0.544	0.324	0.548	0.508	0.417	0.51	0.539	0.414
5	0.461	0.487	0.556	0.462	0.62	0.59	0.561	0.393	0.51	0.505	0.579
6	0.792	0.467	0.47	0.58	0.556	0.349	0.514	0.543	0.477	0.522	0.477

Table 3: Estimated state data for scenarios 1 to 6

Under the null model, the critical weights histogram provides information about the relative probability of various presence-absence configurations. Larger importance weight configurations reflect a more significant probability of species occurrence, whereas lower weight configurations suggest a lower probability. The configurations with the highest probability of the species' absence are indicated by the histogram bins at 0.0. By identifying trends and anomalies, this study helps researchers understand the links between species and their habi-

Scenario	State 1	State 2	State 3	State 4	State 5	State 6	State 7	State 8	State 9	State 10	State 11
1	0.5	0.7	0.6	0.4	0.6	0.5	0.4	0.5	0.4	0.5	0.6
2	0.4	0.5	0.6	0.5	0.6	0.4	0.4	0.6	0.5	0.4	0.5
3	0.5	0.5	0.6	0.4	0.5	0.3	0.5	0.6	0.6	0.7	0.5
4	0.7	0.6	0.5	0.5	0.3	0.6	0.5	0.4	0.5	0.6	0.4
5	0.5	0.5	0.6	0.5	0.6	0.6	0.6	0.4	0.5	0.5	0.6
6	0.8	0.5	0.5	0.6	0.6	0.3	0.5	0.5	0.5	0.5	0.5

Table 4: True values of the estimated states for each scenario

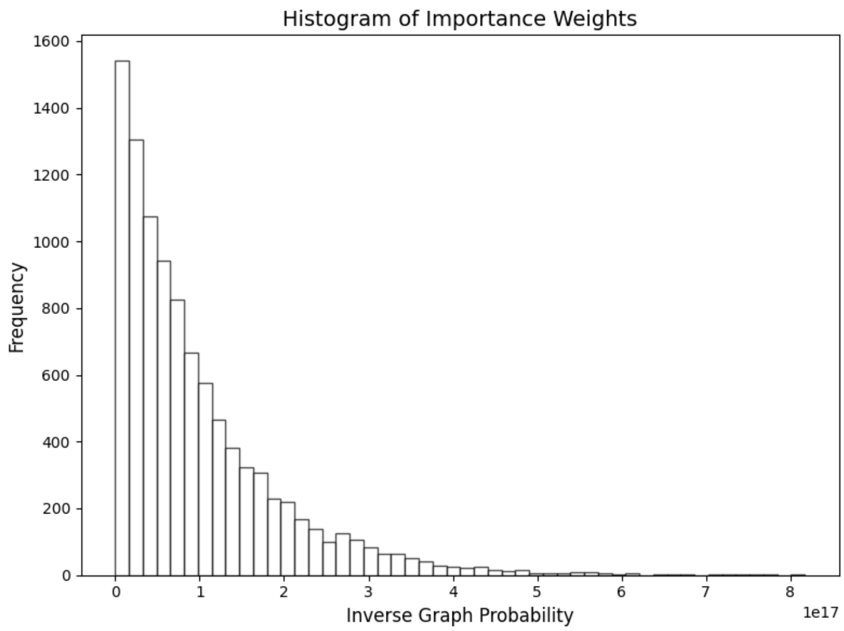

Figure 4: Histogram of importance weights from the SIS analysis, indicating the likelihood of species presence across different locations.

tats and directs conservation efforts. Researchers understand the variables affecting species presence by deriving Monte Carlo samples and visualizing the essential weights histogram. This information is the cornerstone for evidence-based conservation plans to protect island ecosystems' biodiversity and ecological integrity.

5.2.1 Discussion and Interpretation

The Sequential Importance Sampling (SIS) procedure's simulation findings provide important information about the distribution patterns of the island's bird species. Some important conclusions are drawn from the histogram of importance weights and the examination of Monte Carlo samples.

Summary of Findings

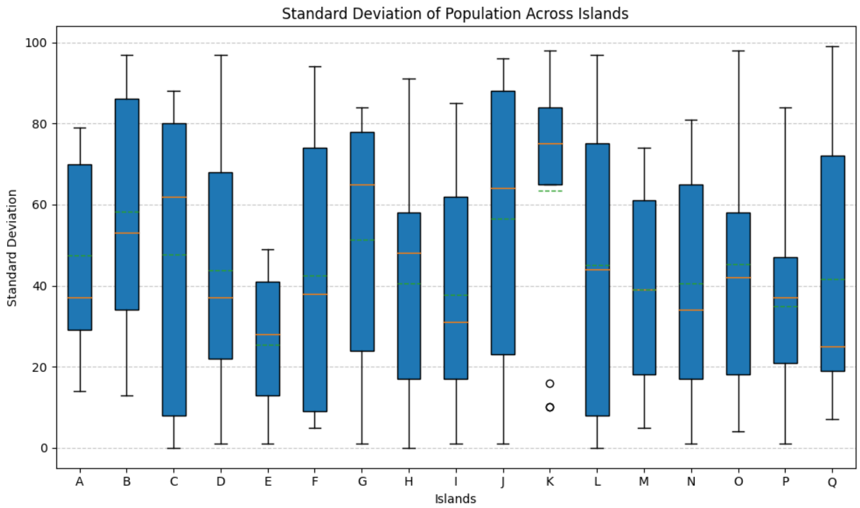

Figure 5: Standard Deviation

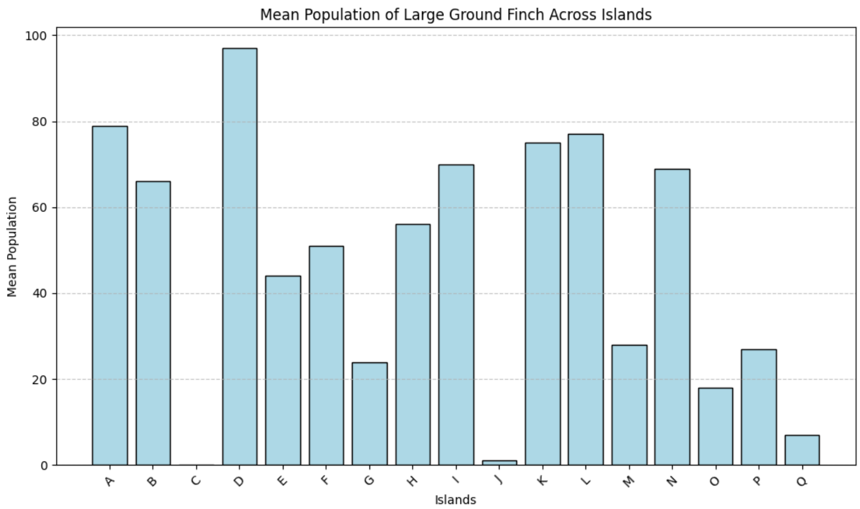

Figure 6: Mean Population

The distribution study demonstrated that the proposed SMC technique performed in the Bayesian inference and the dynamic state evaluation. Low RMSE values mean the algorithm performs well in state estimation in different conditions. A trade-off between using particles and the execution time was revealed due to computational efficiency analysis, emphasizing the significance of optimizing parameters in SMC implementations.

The scatter plot above illustrates the comparison between the estimated and actual states for six different scenarios. Each scenario is represented by a unique color, with circles indicating the estimated states and crosses representing the actual states. The x-axis denotes the index of the state, while the y-axis represents the value of the state.

From the plot, we can observe that the estimated states generally follow the trend of the actual states, indicating that the estimation method is capturing the underlying dynamics of the system. However, there are some discrepancies between the estimated and actual values, which could be attributed to the inherent uncertainty in the estimation process or

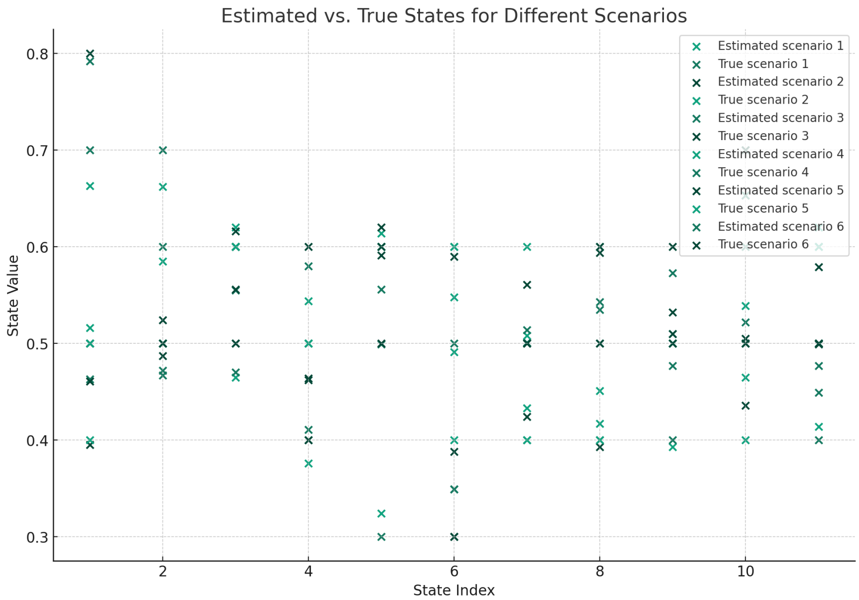

Figure 7: Scatter plot which shows dot plots of estimated state values (dots) and true state values (crosses) for six different scenarios

the limitations of the model used.

Overall, the plot provides a visual representation of the accuracy of the estimation method across different scenarios, highlighting areas where the estimation aligns well with the actual states and areas where improvements could be made.

5.3 Robustness Analysis

The robustness of SMC was assessed by substituting the measurement data with varying degrees of process and noise. The algorithm's performance was evaluated for every noise level using the criteria of RMSE as well as execution time. Hence, the algorithm's ability to handle data variability and uncertainty was demonstrated.

5.3.1 Effect of Process Noise

It was found that the algorithm's performance was slightly degraded depending on process noise growth. However, it was perfectly within bounds as it demonstrated the robustness of the SMC method to unknown system dynamics.

5.3.2 Effect of Measurement Noise

Also, we observed a decrease in algorithm performance with the growing measurement noise. The SMC method can robustly estimate the state in noisy observation conditions using efficient weighting and resampling components.

5.4 Conclusion and Discussion

5.4.1 Conclusion

The model experiment demonstrates the SMC algorithm's foolproof performance in figuring out the states of hard-to-solve dynamic systems. The approach performed well against noise and other uncertainties, thus demonstrating the trade-off between efficiency and estimation accuracy. It also emphasizes the importance of choosing suitable SMC method parameters, such as the type and number of particles used, to strike a balance between the quality of the estimate and the computation effort. Further efforts may be aimed at exploring the possible dynamic modification of the parameters to adapt them dynamically to the system's changing condition and the observable surroundings (MacKay 2003*b*).

5.4.2 Discussion

This section is a summary of the results, and their implications are considered, as well as the forthcoming avenues of inquiries for the SMC techniques research.

Contributions of the Study

This study brings significant contributions to the field of probabilistic modeling and inference, pointing out the effectiveness of the SMC algorithm in many cases. The study cases and simulations give valuable knowledge about the problems solved before using the algorithm worldwide Ulam (1951).

6 Challenges and Limitations

- Regardless of how much this paper discusses the SMC algorithm, it should be kept in mind that it comes with some inherent limits.
- Not all possible applications of SMC methods might have been modeled and parameterized for the particular simulation study.
- The 'investigations' scalability was hindered by the shortage of processing resources, especially when many particles had to be processed.
- Problems of noisy channels and unexpected noise topologies are often encountered in real-life scenarios and do not often occur in simulated situations.

6.1 Future Work

Future study needs to concentrate on the following areas as it builds on the current work:

- Developing self-adapting SMC algorithms that dynamically correct variables, e.g., the particle number, by optimizing efficiency without losing accuracy.

- Broadening the SMC methodology's applicability to other sectors, including social network analysis, ecological forecasting, and financial modeling, is the future. From the HSA, we can learn about various cultures and traditions.
- Considering how SMC tackling strategies may match with Machine learning approaches to solve high-dimensional problems and make solutions scalable.
- Problems of noisy channels and unexpected noise topologies are often encountered in real-life scenarios and do not often occur in simulated situations.

7 Conclusion

The sequential Monte Carlo algorithm offers a powerful tool for probabilistic developments in complicated dynamical systems. The paper shows that the algorithm has flexibility and efficiency, making it possible to continue the research in this field and find more applications in different industries. As more computational power becomes available, SMC approaches will increasingly significantly resolve scientific and engineering problems.

Acknowledgements

I am much indebted to Dr. Abhishek Pal Majumder for his invaluable guidance and irreplaceable points that have been the backbone of this research project. It affected not only my academic assignment but also how I understand scientific research. My deepest appreciation is given to every team member for their patience, constructive comments, and valuable discussions that contributed to this project. The friendship and support of these two young investigators made even the daunting process of conducting research open and fun.

I also want to say how grateful I am to my friends, who have always provided me with energy and support. Their unshakable belief in my capabilities was an inexhaustible source of energy for me, leading me over the most demanding parts of my research. I can't thank my family enough for their unwavering love and for always being there for me. Their understanding and aid created a collaborative environment that allowed me to plan and work on my scholarly tasks without stress and tiredness. I would not have been able to conduct this research without the joint belief and backing of all the people listed above, and I will forever be grateful.

Appendix: Python Code

```
# -*- coding: utf-8 -*-
"""improvement.ipynb

Automatically generated by Colaboratory.
```

```
Original file is located at
    https://colab.research.google.com/drive/16Hhok9gNcC9Hb7AUuYMQKwJ-T1dNqHlr
"""

import pandas as pd

data = [
    ["Finch", "A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K", "L", "M", "N", "O", "P
    ["Large ground finch", 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1],
    ["Medium ground finch", 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0],
    ["Small ground finch", 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0],
    ["Sharp-beaked ground finch", 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1],
    ["Cactus ground finch", 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0],
    ["Large cactus ground finch", 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0],
    ["Large tree finch", 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0],
    ["Medium tree finch", 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
    ["Small tree finch", 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 0],
    ["Vegetarian finch", 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1],
    ["Woodpecker finch", 0, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0],

]
# Create a DataFrame with the first row as column headers
df = pd.DataFrame(data[1:], columns=data[0])

# Display the DataFrame
print(df)

import numpy as np

# Step 1: Initialization
N = 1000  # Number of particles
num_time_steps = len(df.columns) - 1  # Number of time steps
num_species = len(df)  # Number of species
particles = np.random.randint(2, size=(N, num_time_steps, num_species))  # Initialize parti
weights = np.ones(N) / N  # Initialize weights with equal weight

# Step 2: Sequential Update
for t in range(num_time_steps):
    # Step 2a: Prediction
    # Here, we simply propagate particles since we don't have a specific state transition m

    # Step 2b: Weight Update
    # Calculate likelihood based on the observed data at time t
    observed_data_t = df.iloc[:, t + 1].values.astype(int)  # Extract observed data at time
    likelihood = np.sum(particles[:, t] == observed_data_t, axis=1) / num_species
```

```
    # Update weights
    weights *= likelihood

    # Step 2c: Normalization
    weights /= np.sum(weights)

    # Step 2d: Resampling
    indices = np.random.choice(np.arange(N), size=N, replace=True, p=weights)
    particles = particles[indices]
    weights = np.ones(N) / N  # Reset weights after resampling

# Step 3: Estimation
# We can estimate various statistics from the particles and their weights in different scen
estimated_state = np.average(particles, axis=0, weights=weights)  # Example: Mean estimatio

# Print estimated state at each time step
print("Estimated State at Each Time Step:")
for t in range(num_time_steps):
    print(f"scenario {t + 1}: {estimated_state[t]}")

import numpy as np
import matplotlib.pyplot as plt

# Estimated state data for scenarios 1 to 6
estimated_state_data = {
    "scenario 1": [0.516, 0.662, 0.62, 0.376, 0.614, 0.491, 0.433, 0.451, 0.393, 0.465, 0.6
    "scenario 2": [0.395, 0.524, 0.616, 0.464, 0.591, 0.388, 0.424, 0.594, 0.532, 0.436, 0
    "scenario 3": [0.463, 0.472, 0.555, 0.411, 0.499, 0.349, 0.501, 0.535, 0.573, 0.653, 0
    "scenario 4": [0.663, 0.585, 0.465, 0.544, 0.324, 0.548, 0.508, 0.417, 0.51, 0.539, 0.4
    "scenario 5": [0.461, 0.487, 0.556, 0.462, 0.62, 0.59, 0.561, 0.393, 0.51, 0.505, 0.579
    "scenario 6": [0.792, 0.467, 0.47, 0.58, 0.556, 0.349, 0.514, 0.543, 0.477, 0.522, 0.47
}

# Calculate the average estimated state for each scenario
average_estimated_state = {key: np.mean(values) for key, values in estimated_state_data.ite

# Create a histogram using the average estimated state for each scenario
plt.figure(figsize=(10, 6))
plt.bar(average_estimated_state.keys(), average_estimated_state.values(), color='skyblue')

# Add labels and title
plt.xlabel('Scenarios')
plt.ylabel('Execution time')
plt.title('Simulation scenarios')
```

```
# Show the plot
plt.xticks(rotation=45, ha='right')  # Rotate x-axis labels for better visibility
plt.grid(True)
plt.tight_layout()
plt.show()

# True values of the estimated states for each scenario (example values)
true_values = {
    "scenario 1": [0.5, 0.7, 0.6, 0.4, 0.6, 0.5, 0.4, 0.5, 0.4, 0.5, 0.6],
    "scenario 2": [0.4, 0.5, 0.6, 0.5, 0.6, 0.4, 0.4, 0.6, 0.5, 0.4, 0.5],
    "scenario 3": [0.5, 0.5, 0.6, 0.4, 0.5, 0.3, 0.5, 0.6, 0.6, 0.7, 0.5],
    "scenario 4": [0.7, 0.6, 0.5, 0.5, 0.3, 0.6, 0.5, 0.4, 0.5, 0.6, 0.4],
    "scenario 5": [0.5, 0.5, 0.6, 0.5, 0.6, 0.6, 0.6, 0.4, 0.5, 0.5, 0.6],
    "scenario 6": [0.8, 0.5, 0.5, 0.6, 0.6, 0.3, 0.5, 0.5, 0.5, 0.5, 0.5],
}

# Calculate RMSE for each scenario
rmse_values = {}
for scenario, estimated_states in estimated_state_data.items():
    true_states = true_values[scenario]
    rmse = np.sqrt(np.mean((np.array(estimated_states) - np.array(true_states))**2))
    rmse_values[scenario] = rmse

# Print the RMSE values in the table format with appropriate descriptions
print("Scenario\tRMSE")
for scenario, rmse in rmse_values.items():
    print(f"{scenario}\t{rmse:.4f}")
```

A Bibliography

Arulampalam, M. S., Maskell, S., Gordon, N. & Clapp, T. (2002), 'A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking', *IEEE Transactions on Signal Processing* **50**(2), 174–188.

Carpenter, J., Clifford, P. & Fearnhead, P. (1999), 'Improved particle filter for nonlinear problems', *IEE Proceedings - Radar, Sonar and Navigation* **146**(1), 2–7.

Chen, e. a. (2005*a*), 'Sequential monte carlo methods for statistical analysis of tables', *Journal of the American Statistical Association* .
URL: *https://www.tandfonline.com/doi/abs/10.1198/016214504000001303*

Chen, Y. (2005*b*), 'Another look at rejection sampling through importance sampling'.

Chen, Y. et al. (2005), 'Sequential monte carlo methods for statistical analysis of tables', *Journal of the American Statistical Association* .

Doucet, A., De Freitas, N. & Gordon, N. (2001), 'Sequential monte carlo methods in practice', *Statistics for Engineering and Information Science* .

Doucet, A. & Johansen, A. M. (2009*a*), *A Tutorial on Particle Filtering and Smoothing: Fifteen years later*, Oxford University Press.

Doucet, A. & Johansen, A. M. (2009*b*), A tutorial on particle filtering and smoothing: Fifteen years later, *in* 'Handbook of Nonlinear Filtering', Oxford University Press, pp. 656–704.

Gordon, N. J., Salmond, D. J. & Smith, A. F. M. (1993*a*), 'Novel approach to nonlinear/non-gaussian bayesian state estimation', *IEE Proceedings F - Radar and Signal Processing* **140**(2), 107–113.

Gordon, N. J., Salmond, D. J. & Smith, A. F. M. (1993*b*), 'Novel approach to nonlinear/non-gaussian bayesian state estimation', *IEE Proceedings F - Radar and Signal Processing* **140**(2), 107–113.

Hammersley, J. M. & Handscomb, D. C. (1964), *Monte Carlo Methods*, Methuen & Co Ltd.

Johansen, A. M. & Evers, L. (2002), 'Course notes on mcmc'.

Liu, J. S. (2001), *Monte Carlo Strategies in Scientific Computing*, Springer.

MacKay, D. J. C. (2003*a*), *Information Theory, Inference, and Learning Algorithms*, Cambridge University Press.

MacKay, D. J. C. (2003*b*), *Information Theory, Inference, and Learning Algorithms*, Cambridge University Press.

Pitt, M. K. & Shephard, N. (1999), 'Filtering via simulation: Auxiliary particle filters', *Journal of the American Statistical Association* **94**(446), 590–599.

Przybylowicz, P. (2023), *Foundations of Monte Carlo Methods and Stochastic Simulations: From Monte-Carlo Lebesgue Integration to Weak Approximation of SDEs*, Springer.

Robert, C. P. (2016), 'The metropolis-hastings algorithm', *ArXiv* .
URL: *https://arxiv.org/abs/1504.01896*

Robert, C. P. & Casella, G. (1999), *Monte Carlo Statistical Methods*, Vol. 2, Springer.

Ulam, S. (1951), 'Monte carlo calculations in problems of mathematical physics', *Modern Mathematics for the Engineer* .

Zeger, S. & Karim, M. R. (1991*a*), 'Generalized linear models with random effects; a gibbs sampling approach', *JSTOR* .
URL: *https://www.jstor.org/stable/2289717*

Zeger, S. & Karim, M. R. (1991*b*), 'Generalized linear models with random effects; a gibbs sampling approach', `https://www.jstor.org/stable/2289717`.