Sequential Monte Carlo for Applications in Structural Biology, Financial Time Series and Epidemiology

by

Zhaoran Hou

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Statistics

Waterloo, Ontario, Canada, 2024

© Zhaoran Hou 2024

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Member: **Liangliang Wang**
Associate Professor,
Department of Statistics and Actuarial Science,
Simon Fraser University

Supervisor: **Samuel W.K. Wong**
Associate Professor,
Department of Statistics and Actuarial Science,
University of Waterloo

Internal Members: **Martin Lysy**
Associate Professor,
Department of Statistics and Actuarial Science,
University of Waterloo

Glen McGee
Assistant Professor,
Department of Statistics and Actuarial Science,
University of Waterloo

Internal-External Member: **Jangho Yang**
Assistant Professor,
Department of Management Science and Engineering,
University of Waterloo

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

The chapters of this thesis contain Zhaoran Hou's work, under the supervision of Professor Samuel W.K. Wong, which has been submitted to the following peer-reviewed journals.

Chapter 2: Hou, Z. and Wong, S. W.K. (2024). Estimating Boltzmann Averages for Protein Structural Quantities Using Sequential Monte Carlo. *Statistica Sinica*, 34, 1263-1280.

Chapter 3: Hou, Z. and Wong, S. W.K.. Particle Gibbs for Likelihood-Free Inference of State Space Models with Application to Stochastic Volatility. Preprint.

The entirety of the related code is written in `R` and is solely authored by Zhaoran Hou for these manuscripts.

Abstract

Sequential Monte Carlo (SMC) methods are widely used to draw samples from intractable target distributions. Moreover, they have also been adopted to other computational methods for inference such as the particle Markov chain Monte Carlo methods and the SMC^2 methods. In practice, some difficulties arise and are hindering the use of SMC-based methods; examples are the degeneracy of the particles in SMC and the intractability of the target distribution. This thesis addresses these challenges across various domains and proposes effective solutions.

This thesis introduces SMC and SMC-based methods with specific challenges in three diverse fields. Firstly, we propose an SMC method for sampling protein structures from the Boltzmann distribution which is highly constrained, crucial for studying the Boltzmann distribution of protein structures and estimating atomic contacts in viral proteins such as SARS-CoV-2. Secondly, we present a particle Gibbs sampler incorporating the approximate Bayesian computation strategy for stochastic volatility models with intractable likelihoods, offering a solution for parameter inference in financial data analysis by fitting stochastic volatility models to S&P 500 Index time-series data during the 2008 financial crisis. Finally, we introduce a compartmental model with stochastic transmission dynamics and covariates, facilitating better alignment with real-world data for modeling the spread of COVID-19 in Ontario, for which we employ an SMC^2 algorithm incorporating the approximate Bayesian computation strategy.

Acknowledgements

First, I would like to thank Dr. Liangliang Wang, Dr. Martin Lysy, Dr. Glen McGee, and Dr. Jangho Yang very much for their kind willingness to be on my examining committee. As we understood, reading the thesis and attending the PhD defense is quite time-consuming and exhausting, and I appreciate their kindness in reviewing all my PhD work and giving me many insightful comments that can help me improve the thesis to a higher level.

Second, I would like to thank my supervisor, Dr. Samuel Wong, for his meticulous help with my research during my PhD life, especially when everything was tough during the pandemic. Throwback to May 2019 when I was searching for a supervisor for PhD, and luckily Samuel was one of the few professors who were still open to new students. As his first PhD student graduating from Waterloo, I'm grateful to have him as my supervisor in this unforgettable experience in my life.

Next, I would like to thank Dongrui Bai, Jianchu Chen, Jingyue Huang, Mingzhe Huang, Bingying Xia, Tianzhu Yang, Chi-Kuang Yeh and Liyuan Zheng (alphabetically ordered by last names) who have given me much help throughout my PhD journey; moreover, I also want to thank Daoyi Chen, Ying Chi, Haoyuan Dong, Ruoyu Du, Zhaoyu Guo, Kecheng Li, Pusheng Li, Wei Liang, Yechao Meng, Sheng Wang, Ziyu Wang, Zeyu Wei, Jiachen Wu, Hedi Xia, Yunjie Ying, Qihuang Zhang and Xisi Zhang (alphabetically ordered by last names) who ever supported me emotionally and I could not be who I am without them.

Last, I want to thank my family for their constant support. It is not easy for all of us to finish the journey in academia as there have been many difficulties; fortunately, I finally grew up enough to leave for industry from school.

Table of Contents

List of Figures x

List of Tables xiii

1 Introduction 1

1.1 Filtering and Smoothing . . . 2

1.2 Parameter Learning . . . 4

1.3 Sampling from a General Multivariate Distribution . . . 7

1.4 Summary and Contribution . . . 8

2 Estimating Boltzmann Averages for Protein Structural Quantities Using Sequential Monte Carlo 12

2.1 Introduction . . . 12

2.2 Motivating Application: Estimating Protein Structural . . . 15

2.3 Methodology . . . 16

2.3.1 Review of Fearnhead and Clifford's SMC method . . . 16

2.3.2 A general upsampling-downsampling SMC framework . . . 17

2.4 Estimating Protein Structural Quantities . . . 20

2.5 Simulation study . . . 23

2.6 Example: Estimating the atomic contacts in the SARS-CoV-2 spike protein 26

2.7 Conclusion and Discussion . . . 28

3 Particle Gibbs for Likelihood-Free Inference of Stochastic Volatility Models **29**

3.1 Introduction to Financial Time Series and Volatility Modeling 29

3.2 Methodology 34

3.2.1 Model setup 34

3.2.2 Particle Gibbs for the SVM with tractable likelihoods 35

3.2.3 Review of ABC methods and ABC-SMC methods 37

3.2.4 Likelihood-free ABC-based cSMC for the SVM 38

3.3 Simulation study 42

3.4 Application: the S&P 500 Index during the Financial Crisis in 2008 44

3.5 Conclusion and Discussion 44

4 Inference on a Markovian Compartmental Model for COVID-19 in Ontario **51**

4.1 Introduction to compartmental models in epidemiology 51

4.2 Methodology 55

4.2.1 Model setup 55

4.2.2 The ABC-SMC^2 method 56

4.3 Simulation study 59

4.4 Discussion 61

5 Conclusion and Future Work **66**

References **67**

APPENDICES **79**

A Supplementary Material for Chapter 2 **80**

A.1 Pseudocode for SISR algorithm 80

A.2 Proper weighting of the proposed sampling scheme 81

A.3 Minimization of the conditional expected squared error loss 83
A.4 Details of the energy function 84
A.5 UDSMC experiment for different values of M 86
A.6 Pseudocode for SISR with one-step lookahead 88
A.7 Particle diversity of UDSMC and SISR 88

B Supplementary Material for Chapter 3 **92**
B.1 Proof of Proposition 3 92
B.2 Impact of a smaller ϵ in the ABC kernel 93

C Supplementary Material for Chapter 4 **97**
C.1 Trajectory plots generated with one of the 100 estimates 97

List of Figures

1.1 The plot of the number of the distinct particles over iteration times through the bootstrap filter. For simplicity, the state space of the state x_t is $\{1, 2, \ldots, 100\}$ with uniform prior and the weight of $x_{1:T}$ is proportional to $\prod_{t=1}^{T} \exp(x_t/10)$. With 100 different starting values as the initial value for each particle, the diversity quickly drops to around 30. 10

1.2 The plot of the number of different parameters evolving along with the hidden states in the bootstrap filter. $\{x_0^{(n)} = 0\}_{n=1}^{100}$ each with $w_0^{(n)} = n$, we assume $x_{t+1} \mid x_t \sim N(x_t, 1)$, $y_t \mid x_t, \omega \sim N(x_t, \omega^2)$ and $y_t = 0$ for all t. The number of different parameters starts from 100, drops below 10 after several iterations and ends up with 1. 11

2.1 Illustration of connections and dihedral angles in a protein segment. Solid lines in the figure indicate bonds (connections). Within the amino acid indexed by t, N^t, C_α^t, C^t and O^t are successively connected, and the side chain R^t (that distinguishes different amino acid types) is connected to C_α^t; the double lines between C^t and O^t indicate a double-bond connection. Successive amino acids are connected via a C-N bond, e.g., amino acids indexed by t and $t+1$ are connected with the bond between C^t and N^{t+1}. To define the dihedral angle ϕ_t, consider the two planes marked in the figure: Plane A (red) consists of atoms C^{t-1}, N^t and C_α^t; Plane B (blue) consists of atoms N^t, C_α^t and C^t. Then ϕ_t is the angle between Plane A and Plane B when looking down the $\mathrm{N} - \mathrm{C}_\alpha$ bond. 21

2.2 Estimated Boltzmann averages for the atomic contacts of C_α atoms at amino acid positions 473 to 489 of the SARS-CoV-2 spike protein, using the samples obtained from running UDSMC with $N = 500000$ and $M = 20$. 27

3.1 The normal Q-Q plot of the S&P 500 Index returns from January 2008 to March 2009. Both tails of the returns are much heavier than the Gaussian tails, which implies the Gaussian assumption for Z_t in Equation (3.1) is not proper and needs to be replaced with a more heavy-tailed distribution. . . 32

3.2 The left panel presents the daily returns of the S&P 500 index from January 2008 to March 2009. The large fluctuations around October 2008 indicate the climax of the global financial crisis. The 2.5% and 97.5% quantiles of the sampled returns are shown with red dashed lines and the corresponding 95% credible interval for the returns is highlighted in red. The right panel presents the fitted daily volatility (black line) with a 95% credible interval (red dashed lines); the high volatility around the climax is well-captured. . 45

4.1 The boxplot of the 100 estimates produced from ABC-SMC^2 for each parameter with kernel H defined in (i). The true values are set as $\beta_{0,SE} = \beta_{1,SE} = \beta_{2,SE} = 0.5$, $\beta_{EI} = 1$, $\beta_{IR} = 2$ and $\sigma_{SE} = \sigma_{EI} = \sigma_{IR} = 0.01$, and the prior are set as all β's follow $N(0, 1)$ and all σ's follow $\Gamma(0.2, 0.5)$. . . . 63

4.2 The boxplot of the 100 estimates produced from ABC-SMC^2 for each parameter with kernel H defined in (ii). The true values are set as $\beta_{0,SE} = \beta_{1,SE} = \beta_{2,SE} = 0.5$, $\beta_{EI} = 1$, $\beta_{IR} = 2$ and $\sigma_{SE} = \sigma_{EI} = \sigma_{IR} = 0.01$, and the prior are set as all β's follow $N(0, 1)$ and all σ's follow $\Gamma(0.2, 0.5)$. . . . 64

4.3 The boxplot of the 100 estimates produced from ABC-SMC^2 for each parameter with kernel H defined in (iii). The true values are set as $\beta_{0,SE} = \beta_{1,SE} = \beta_{2,SE} = 0.5$, $\beta_{EI} = 1$, $\beta_{IR} = 2$ and $\sigma_{SE} = \sigma_{EI} = \sigma_{IR} = 0.01$, and the prior are set as all β's follow $N(0, 1)$ and all σ's follow $\Gamma(0.2, 0.5)$. . . . 65

A.1 Values of ϕ_s (left panels) and ψ_s (right panels) among resampled particles, for $s = 0, 1, 2, 3$ after each iteration (t) of UDSMC (blue) and SISR (red), for the protein segment in the simulation study. 90

A.2 Values of ϕ_s (left panels) and ψ_s (right panels) among resampled particles, for $s = 4, 5, 6, 7$ after each iteration (t) of UDSMC (blue) and SISR (red), for the protein segment in the simulation study. 91

C.1 The four figures demonstrate the simulated trajectories of the four compartments using the kernel H defined in (i) in Section 4.3 with respect to a set of randomly-picked estimates for the parameters from the 100 repetitions. Three trajectories simulated from the ABC-SMC using the estimates are presented in each figure, including the one with the largest particle weight, the one with the smallest particle weight, and the mean trajectory. 98

C.2 The four figures demonstrate the simulated trajectories of the four compartments using the kernel H defined in (ii) in Section 4.3 with respect to a set of randomly-picked estimates for the parameters from the 100 repetitions. Three trajectories simulated from the ABC-SMC using the estimates are presented in each figure, including the one with the largest particle weight, the one with the smallest particle weight, and the mean trajectory. 99

C.3 The four figures demonstrate the simulated trajectories of the four compartments using the kernel H defined in (iii) in Section 4.3 with respect to a set of randomly-picked estimates for the parameters from the 100 repetitions. Three trajectories simulated from the ABC-SMC using the estimates are presented in each figure, including the one with the largest particle weight, the one with the smallest particle weight, and the mean trajectory. 100

List of Tables

2.1 Boltzmann averages of the eight quantities of interest. The 10^8 samples obtained using our brute force approach were used as a proxy for the ground truth. 24

2.2 RMSEs of the eight quantities, based on 100 repetitions of UDSMC ($M = 20$), SISR and lookahead SISR ($L = 20$) for the protein segment $\mathbf{x}_{284:291}$. The corresponding rows of each method have a similar computational cost: e.g., UDSMC with $N = 5000$ and $M = 20$, SISR with $N = 100,000$, and lookahead SISR with $N = 5000$ and $L = 20$. 25

3.1 RMSEs of the SVM parameter estimates for the three particle Gibbs algorithms, based on 100 simulated datasets with $T = 100$, $N = 100$, $Z_t \sim SD(1.75, 0.1, 1, 0)$, $\epsilon = 0.001$. 47

3.2 RMSEs of the SVM parameter estimates for the three particle Gibbs algorithms, based on 100 simulated datasets with $T = 100$, $N = 100$, $Z_t \sim SD(1.7, 0.3, 1, 0)$, $\epsilon = 0.001$. 48

3.3 RMSEs of the SVM parameter estimates for the three particle Gibbs algorithms, based on 100 simulated datasets with $T = 100$, $N = 100$, $Z_t \sim SD(1.5, -0.3, 1, 0)$, $\epsilon = 0.001$. 49

3.4 Estimates and credible intervals of τ, ϕ and σ_h^2 based on fitting an SVM to S&P 500 Index data from January 2008 to March 2009. The numerical results were obtained from the ABC-PG-cAPF algorithm with $N = 500$, $Z_t \sim SD(1.725, 0.0915, 1, 0)$, $\epsilon = 0.001$, and 5000 posterior samples. 50

4.1 A summary of the inference results of ABC-SMC^2 for different propagation kernels H in the PMCMC kernel. The bias (top) and RMSE (bottom) computed with the estimates from the 100 repetitions are reported for each parameter. The true values are set as $\beta_{0,SE} = \beta_{1,SE} = \beta_{2,SE} = 0.5$, $\beta_{EI} = 1$, $\beta_{IR} = 2$ and $\sigma_{SE} = \sigma_{EI} = \sigma_{IR} = 0.01$, and the prior are set as all β's follow $N(0, 1)$ and all σ's follow $\Gamma(0.2, 0.5)$. 61

A.1 Variance of the UDSMC estimates over 100 repetitions when estimating the Boltzmann average of the atomic contacts of C_α at each position from 283 to 292, for different combinations of NM and M. "NA" indicates that the algorithm failed to complete for that combination of N and M. 87

B.1 RMSEs of the SVM parameter estimates for the three particle Gibbs algorithms, based on 100 simulated datasets with $T = 100$, $N = 100$, $Z_t \sim SD(1.75, 0.1, 1, 0)$, $\epsilon = 0.0005$. 94

B.2 RMSEs of the SVM parameter estimates for the three particle Gibbs algorithms, based on 100 simulated datasets with $T = 100$, $N = 100$, $Z_t \sim SD(1.7, 0.3, 1, 0)$, $\epsilon = 0.0005$. 95

B.3 RMSEs of the SVM parameter estimates for the three particle Gibbs algorithms, based on 100 simulated datasets with $T = 100$, $N = 100$, $Z_t \sim SD(1.5, -0.3, 1, 0)$, $\epsilon = 0.0005$. 96

Chapter 1

Introduction

Sequential Monte Carlo (SMC) methods, also known as particle filters, are a flexible class of simulation-based Monte Carlo algorithms for sampling from a target distribution. SMC methods originated from the on-line inference problems in dynamical systems, where observations arrive sequentially and interest lies in the posterior distribution of hidden state variables (Liu and Chen, 1998). The key idea of SMC is to represent the target distribution with a set of weighted particles that are updated incrementally (Del Moral, 1997). This sequentially-updating framework has been applied to a wide range of problems such as target tracking (Hue et al. (2002); Moghaddasi and Faraji (2020)), time series analysis (Ping et al. (2018); Urteaga et al. (2017)), and computer vision (Vermaak et al., 2001). Moreover, SMC has increasingly found utility in general multivariate settings beyond dynamical systems, such as graphical models (Paige and Wood, 2016), population genetics (Smith et al., 2017), and graph matching (Jun et al., 2019).

Hidden Markov models (HMMs) and state-space models (SSMs) are provided as concrete examples to review SMC concepts. HMMs are a class of models in which the observed data are produced from a hidden system which is a Markov process. This class of models has been applied in many fields such as computational finance (Petropoulos et al. (2016) and Sipos et al. (2017)), speech recognition (Domingos, 2015), protein folding (Stigler et al., 2011), DNA motif discovery (Wong et al., 2013) and transportation forecasting (Zarwi et al., 2017). Consider the HMM defined in Doucet et al. (2001), which consists of a sequence of unobserved one-dimensional signals or states $\{x_t \in \mathcal{X} \mid t = 0, 1, 2, \dots\}$ and observations $\{y_t \mid t = 1, 2, \dots\}$. The HMM can be characterized by (i) the initial state distribution $p(x_0)$; (ii) the transition probability $p(x_{t+1} \mid x_t)$; (iii) the observation probability

$p(y_t \mid x_t)$ for $t \geq 1$, and it can be graphically represented as

$$\begin{array}{ccccccccc} x_0 & \rightarrow & x_1 & \rightarrow & x_2 & \rightarrow & x_3 & \rightarrow \cdots \rightarrow & x_T \\ & & \downarrow & & \downarrow & & \downarrow & & \downarrow \\ & & y_1 & & y_2 & & y_3 & & y_T \end{array}$$

where the arrows indicate probabilistic dependence. The state space $\mathcal{X}$ in HMMs is assumed finite, while the cases where $\mathcal{X}$ is continuous are known as SSMs. In our context, we follow these traditional definitions for the two classes of models. There are two types of problems for HMMs/SSMs: (i) the filtering and smoothing problems; (ii) the parameter learning problems. There have been many algorithms designed to solve the two problems, e.g. expectation-maximum (EM) algorithms proposed by Dempster et al. (1977) and adapted by Xuan et al. (2001) and Cappé (2011), some variants of likelihood methods such as Fridman and Harris (1998) and Sandmann and Koopman (1998) and Markov chain Monte Carlo (MCMC) methods such as Jacquier et al. (2002). Particularly, SMC methods were initially developed for HMMs/SSMs and can be applied in models with more general settings, and these concepts will be discussed in the following subsections.

1.1 Filtering and Smoothing

The filtering problem aims to estimate $p(x_t \mid y_{1:t})$ (sometimes $p(x_{0:t} \mid y_{1:t})$) where $t < T$ (with parameters given, if any) while the smoothing problem aims to estimate $p(x_t \mid y_{1:T})$ (sometimes $p(x_{0:T} \mid y_{1:T})$). SMC methods were initially designed for solving filtering and smoothing, and the bootstrap filter of Gordon et al. (1993) is considered to be the first SMC algorithm, which is still commonly used for its simplicity in propagation and resampling (Fearnhead and Clifford, 2003). Its framework is summarized as:

1. Sample $\{x_0^{(n)}\}_{n=1}^N$ from $p(x_0)$.
2. Sample $\{\tilde{x}_{t+1}^{(n)}\}_{n=1}^N$ from $p(x_{t+1} \mid x_t^{(n)})$ and set $\tilde{x}_{0:t+1}^{(n)} = (x_{0:t}^{(n)}, \tilde{x}_{t+1}^{(n)})$ for $t = 0, 1, \ldots, T$.
3. Evaluate the importance weights $\tilde{w}_{t+1}^{(n)} = p(y_{t+1} \mid \tilde{x}_{t+1}^{(n)})$ for each $\tilde{x}_{t+1}^{(n)}$.
4. Resample N particles $\{x_{0:t+1}^{(n)}\}_{n=1}^N$ with replacement from $\{\tilde{x}_{0:t+1}^{(n)}\}_{n=1}^N$ proportional to the importance weights.
5. Go back to step 2 until the last observation is reached.

The bootstrap filter samples one descendant for each particle in the propagation step and resamples from the propagated particles with replacement in the resampling step. As a feature of the bootstrap filter, it operates the propagation step before the resampling step and thus is known as a propagate-resample filter. The bootstrap filter often produces less distinct samples, known as particle weight degeneracy.

Particle weight degeneracy is a common problem in implementing SMC. For clarity of exposition, Figure (1.1) presents an example. Assume the HMM structure and a bootstrap filter with $T = N = 100$ and 100 distinct starting particles as $\{x_0^{(n)}\}_{n=1}^{100}$, the number of distinct particles drops to a low level quickly with a simple transition probability and a simple observation probability. We obtain 100 particles with many repetitions from the bootstrap filter after 100 iterations, which leads to a large variance in the Monte Carlo estimates. Therefore, we want to avoid weight degeneracy for stabler estimates when implementing SMC.

In contrast, another typical particle filter, the auxiliary particle filter (APF) proposed by Pitt and Shephard (1999), achieves the same goal as the bootstrap filter does and avoids particle decay by operating in the order of resampling and propagation to produce more evenly-weighted particles. The auxiliary particle filter uses a different representation of the joint filtering distribution of (x_t, x_{t+1}) as

$$\begin{aligned} p(x_t, x_{t+1} \mid y_{1:t+1}) &\propto p(x_{t+1} \mid x_t, y_{1:t+1}) p(x_t \mid y_{1:t+1}) \\ &= p\left(x_{t+1} \mid x_t, y_{1:t+1}\right) p\left(y_{t+1} \mid x_t\right) p\left(x_t \mid y_{1:t}\right). \end{aligned}$$

Pitt and Shephard (1999) proposed an importance function $p(y_{t+1} \mid \hat{\mu}_{t+1} = g(x_t))$ for the adaption feature of their method, i.e. the step to propagate states takes into account of the observations as well. For example, given the current particles $\{x_t^{(n)}\}_{n=1}^N$, $g(x_t)$ can be chosen as the mean, the mode, a draw or any other value associated with $p(x_{t+1} \mid x_t^{(n)})$. Then, the steps of the auxiliary particle filter are:

1. Resample $\{\tilde{x}_t^{(n)}\}_{n=1}^N$ from $\{x_t^{(n)}\}_{n=1}^N$ proportional to $p(y_{t+1} \mid g(x_t^{(n)}))$.
2. Propagate $\{\tilde{x}_t^{(n)}\}_{n=1}^N$ to $\{\tilde{x}_{t+1}^{(n)}\}_{n=1}^N$ via $p(x_{t+1} \mid \tilde{x}_t^{(n)})$.
3. Resample $\{\tilde{x}_{t+1}^{(n)}\}_{n=1}^N$ proportional to $p(y_{t+1} \mid \tilde{x}_{t+1}^{(n)}) / p(y_{t+1} \mid g(\tilde{x}_t^{(n)}))$.

The introduced methods share a feature that they sample one descendant in the propagation step, so it is possible to have a few dominant particles with much higher weights

than the rest which leads to particle weight degeneracy. In contrast, Fearnhead and Clifford (2003) proposed an SMC framework for HMMs that implements an exhaustive search over the finite state space. They emphasized two important features of their method: (i) the method produces unbiased particle weights which can then be used for Monte Carlo estimation; (ii) the produced particle weights minimize the squared error loss function, so the downsampling scheme guarantees the smallest variance of the Monte Carlo estimates. These two features are important to compute the Monte Carlo integration, whether for discrete or continuous state spaces. Their SMC strategy inspired us to develop its extension for continuous spaces; in Chapter 2, we present our generalization of this SMC strategy of Fearnhead and Clifford (2003) for continuous spaces for sampling protein structures. Suppose we sample multiple descendants, e.g., M, during propagation for each of the N particles. Even though we cannot explore all possible candidates in continuous space, multiple descendants allow us to probe more of the space, such that the resampled N particles from the MN particles can be more likely to concentrate on the modes of the target distribution and have more uniform weights.

Sometimes HMMs/SSMs can adopt intractable likelihood functions for the observations. We consider this setting in financial time series modeling in Chapter 3. The intractable likelihood can hinder the filtering/smoothing problems as it is impossible to evaluate (or expensive to accurately approximate) $p(y_t \mid x_t)$. In this case, we shall consider the strategy of approximate Bayesian computation (ABC, Marin et al., 2012): for a candidate x_t we generate a sample y_t^* from the intractable likelihood and assign a weight to x_t based on how close the generated sample is to the observed y_t. The SMC methods adopting ABC are known as ABC-SMC in our context.

1.2 Parameter Learning

Given the settings of HMMs/SSMs, the transition probability $p(x_{t+1} \mid x_t)$ and the observation probability $p(y_t \mid x_t)$ can also depend on some parameters ω. For example, we have noisy autoregressive models with unknown coefficients in time series. If the parameters ω involved in $p(x_{t+1} \mid x_t, \omega)$ and $p(y_t \mid x_t, \omega)$ are unknown and need to be estimated, it is known as the parameter learning problem. Parameter learning is, compared to filtering and smoothing, more challenging for SMC. A naive method is that if we initially assign values of ω to each particle, i.e., $\omega_0^{(n)}$ to $x_0^{(n)}$ for $n = 1, \ldots, N$, let ω evolve with $x_{0:t}$'s through the bootstrap filter, i.e., $(\omega_{t+1}^{(n)}, x_{0:t+1}^{(n)}) = (\omega_{t+1}^{(n)}, x_{0:t}^{(n)}, x_{t+1}) = (\omega_t^{(n)}, x_{0:t}^{(n)}, x_{t+1})$, where $x_{0:t}^{(n)}$ is propagated through the bootstrap filter with the corresponding $\omega_t^{(n)}$. However, it can lead to particle decay if simply letting ω evolve along with states x's so the diversity of particles

decreases over time and the target distribution $p(x_{0:T}, \omega \mid y_{1:T})$ is poorly represented by the samples (Carvalho et al., 2010). Figure 1.2 presents an example: given 100 initial states $\{x_0^{(n)} = 0\}_{n=1}^{100}$ each with $w_0^{(n)} = n$, we assume $x_{t+1} \mid x_t \sim N(x_t, 1)$, $y_t \mid x_t, \omega \sim N(x_t, \omega^2)$ and $y_t = 0$ for all t. The results indicate that the diversity of the ω's evolving along with the states vanishes as there is only one value remaining for ω after fifteen iterations, i.e., we obtain only one sample for ω through the bootstrap filter. SMC methods designed for parameter learning and avoid particle decay are limited, and there are two fundamental SMC algorithms for parameter learning: the Storvik's filter (Storvik, 2002) and the Liu and West's filter (Liu and West, 2001).

Storvik (2002) assumes that $p(\theta \mid x_{1:t}, y_{1:t})$ depends on a set of sufficient statistics defined by $s_t = \mathcal{S}(s_{t-1}, x_t, y_t)$, which are able to be updated recursively in iterations, and $q(\cdot)$ is a proposed distribution conditional on y_{t+1} leading to the adaption of this particle filter. The Storvik's filter can be summarized as: (i) propagate $\{x_t^{(n)}\}_{n=1}^N$ to $\{\tilde{x}_{t+1}^{(n)}\}_{n=1}^N$ via $q(x_{t+1} \mid x_i^{(n)}, \theta^{(n)}, y_{1:t+1})$; (ii) resample $\{(x_{t+1}, s_i)^{(n)}\}_{n=1}^N$ from $\{(\tilde{x}_{t+1}, s_i)^{(n)}\}_{n=1}^N$ proportional to $p(y_{t+1} \mid \tilde{x}_{t+1}^{(n)}, \theta) p(\tilde{x}_{t+1}^{(n)} \mid x_t^{(n)}, \theta) / q(\tilde{x}_{t+1}^{(n)} \mid x_t^{(n)}, \theta, y_{1:t+1})$; (iii) propagate sufficient statistics $s_{t+1}^{(n)} = \mathcal{S}(s_t^{(n)}, x_{t+1}^{(n)}, y_{t+1})$; (iv) sample $\theta^{(n)}$ from $p(\theta \mid s_{t+1}^{(n)})$.

Rather than involving sufficient statistics, the Liu and West's filter approximates $p(\theta \mid y_{1:t})$ with a mixture of normals. Assume a set of particles $\{(x_t, \theta_t)^{(n)}\}_{n=1}^N$ from $p(x_t, \theta \mid y_{1:t})$, and the posterior $p(\theta \mid y_{1:t})$ can be approximated by the mixture distribution

$$\sum_{n=1}^{N} N\left(\mu^{(n)}; h^2 V_t\right),$$

where $\mu^{(n)} = a\theta_t^{(n)} + (1-a)\bar{\theta}_t$, $\bar{\theta}_t = \sum_{n=1}^N \theta_t^{(n)}/N$ and $V_t = \sum_{n=1}^N (\theta_t^{(n)} - \bar{\theta}_t)(\theta_t^{(n)} - \bar{\theta}_t)'/N$. The constants a and h represent the extent of shrinkage and overdispersion of the mixture respectively. Similar to the auxiliary particle filter, Liu and West's filter involves the map $g(\cdot)$ and adopts a resample-propagate framework. Both the mixture approximation and the resample-propagate framework are used to avoid particle decay when they generate samples from the current posterior (see details in Liu and West (2001)).

The two introduced filters solve filtering and parameter learning together as t increases, and they produce the samples $\{(x_T^{(n)}, \omega_T^{(n)})\}_{n=1}^N$ once reaching the last observation y_T. With the samples $\{(x_T^{(n)}, \omega_T^{(n)})\}_{n=1}^N$, the forward filtering backward smoothing (FFBS) framework is usually implemented for smoothing, i.e., estimation of the distribution of $(x_{0:T}, \omega)$ given $y_{1:T}$. Forward filtering indicates that $p(x_t, \omega \mid y_{1:t})$ are iteratively estimated; once $t = T$ reached, with $p(x_{t+1} \mid x_t, \omega)$, $p(y_t \mid x_t, \omega)$ and the last estimates of x_t, the previous states

can be estimated backward from $t = T-1$ to $t = 0$, so that we can obtain $\{(x_{0:T}^{(n)}, \omega_T^{(n)})\}_{n=1}^N$ as smoothing results.

We have introduced two fundamental SMC methods for parameter learning, which have also been extended by other researchers, such as Carvalho et al. (2010) and Yang et al. (2018). The SMC of Storvik (2002) and its variants involve the parameters based on a set of sufficient statistics, which are derived from the conjugate priors. Therefore, it confines the choices for the prior and the form of the transition distribution. For example, in Bayesian linear regression, a conjugate prior of the normal-inverse-gamma distribution requires the normality of the noise. However, if a non-conjugate prior is applied or the noise is assumed to be non-Gaussian, e.g., the student-t distribution, the methods of sufficient statistics are no longer available. Liu and West (2001) instead approximate the marginal distribution of the parameters with a mixture normal distribution. However, the kernel mixture approximation, compared to using sufficient statistics, can lead to greater Monte Carlo errors due to the high-dimensional parameter space (Rios and Lopes, 2013). Therefore, those SMC methods do not generally solve the parameter learning problem and may need to be extended depending on the specific application.

In addition to SMC methods, there are also MCMC methods designed for this parameter learning problem. For example, Jacquier et al. (2002) proposed a pure MCMC method for the classic stochastic volatility model that approximates the univariate density of the volatility with an inverse-gamma variable and draws samples of both the states and parameters using a Gibbs method; Raggi and Bordignon (2008) proposed a parameter learning algorithm based on the auxiliary particle filter together with the MCMC methodology to reduce the correlation between particles; Andrieu et al. (2010) proposed a combined use of MCMC and SMC, pioneering a class of algorithms known as particle MCMC (PMCMC), for filtering and parameter estimation where SMC is used for smoothing within an MCMC algorithm to approximate the weights of parameters.

When tackling HMMs/SSMs with intractable likelihood such as the stochastic volatility model for financial time series introduced in Chapter 3, PMCMC can adopt an ABC-SMC algorithm and thus is known as ABC-PMCMC. In particular, when a Gibbs-sampler structure is considered in PMCMC, which is known as particle Gibbs, it needs to incorporate a variant class of SMC methods, known as conditional SMC (cSMC), which preserves a trajectory intact throughout the propagation-resampling procedure (Andrieu et al., 2010). Similar to SMC methods, cSMC methods can also adopt ABC settings when tackling intractable likelihood, which we call ABC-cSMC in our context. In Chapter 3, we present the use of different types of ABC-cSMC samplers within a particle Gibbs algorithm for modeling stochastic volatility in financial time series.

1.3 Sampling from a General Multivariate Distribution

SMC can also be used to sample from a general multivariate distribution $p(\mathbf{x}_{0:T})$ that does not necessarily have the Markovian structure. Removing the Markovian assumption can complicate the problem since it becomes more expensive to evaluate $p(\mathbf{x}_t \mid \mathbf{x}_{0:t-1})$ as t increases. In this case, the fundamental principle of SMC is to decompose the joint distribution as

$$p(\mathbf{x}_{0:T}) = p(\mathbf{x}_0)p(\mathbf{x}_1 \mid \mathbf{x}_0)p(\mathbf{x}_2 \mid \mathbf{x}_{0:1}) \cdots p(\mathbf{x}_T \mid \mathbf{x}_{0:T-1}),$$

and

$$p(\mathbf{x}_{0:t+1}) = p(\mathbf{x}_{0:t}) \cdot p(\mathbf{x}_{t+1} \mid \mathbf{x}_{0:t}).$$

This equation allows us to approximate $p(\mathbf{x}_{0:T})$ step by step via approximating the intermediate joint distribution $p(\mathbf{x}_{0:t+1})$ using SMC until obtaining completed sequences. For example, given the weighted particles approximating $p(\mathbf{x}_{0:t})$, SMC can produce a new set of weighted particles to approximate $p(\mathbf{x}_{0:t+1})$ via sequential importance sampling (SIS) scheme over those particles and eventually generates N weighted sequences of state $\{\mathbf{x}_{0:T}^{(n)}\}_{n=1}^N$ to represent the density of the true target distribution $p(\mathbf{x}_{0:T})$. Assume a set of generated particles $\{\mathbf{x}_{0:t}^{(n)}\}_{n=1}^N$ with weights $\{w_t^{(n)}\}_{n=1}^N$, then for each $\mathbf{x}_{0:t}^{(n)}$, we sample a descendant $\mathbf{x}_{t+1}^{(n)}$ from a proposal density function $\eta_{t+1}(\mathbf{x}_{t+1} \mid \mathbf{x}_{0:t}^{(n)})$ and define $\mathbf{x}_{0:t+1}^{(n)} = (\mathbf{x}_{0:t}^{(n)}, \mathbf{x}_{t+1}^{(n)})$ with a weight

$$w_{t+1}^{(n)} \propto w_t^{(n)} \cdot \frac{p(\mathbf{x}_{t+1}^{(n)} \mid \mathbf{x}_{0:t}^{(n)})}{\eta_{t+1}(\mathbf{x}_{t+1}^{(n)} \mid \mathbf{x}_{0:t}^{(n)})}.$$

In Chapter 2, we aim to sample protein structures using SMC, and the continuous states $\mathbf{x}_{0:T}$ are the geometric degrees of freedom, i.e., dihedral angles, that are sufficient to determine the coordinates of the backbone atom. Then, the target distribution $p(\mathbf{x}_{0:T})$ is obtained according to the potential energy of the structures. This setup does not have the Markovian property (and is not an HMM/SSM) as the current state $\mathbf{x}_t$ depends on all previous states $\mathbf{x}_{0:t-1}$, rather than the last, and thus provides an example of using SMC to sample from a general multivariate distribution. Moreover, weight degeneracy is severe in this problem as the support of the target distribution is highly constrained due to geometric constraints induced by the protein energy evaluation. We shall show that adapting the strategy of Fearnhead and Clifford (2003) can help mitigate this weight degeneracy.

Sampling from the multivariate distribution $p(\mathbf{x}_{0:T})$ can also be an important part of parameter learning, and can be incorporated with parameter learning algorithms such as MCMC or PMCMC as presented in Chapter 4. One difficulty arises when $p(\mathbf{x}_{0:T})$ is Markovian but has an intractable transition probability $p(\mathbf{x}_{t+1} \mid \mathbf{x}_t)$; for example, in Chapter 4, we present a compartmental model that has an intractable transition probability between compartments at successive time points; thus, the smoothing of this problem also requires the use of approximate Bayesian computation (ABC) to bypass evaluating the intractable probability. It is natural to consider applying ABC-PMCMC for parameter learning; however, ABC-based particle Gibbs methods are not implementable due to the lack of access to closed-form conditional distributions. The ABC-based particle Metropolis-Hastings algorithm (Andrieu et al., 2010) seems to be the only implementable ABC-PMCMC method. Still, we find that it is ineffective in practice since the MCMC chain can easily get stuck in local modes, which slows down its convergence. Therefore, we propose to use an SMC framework that incorporates an ABC-SMC and an ABC-PMCMC for parameter learning, known as ABC-SMC^2 methods (Chopin et al., 2013; Kerama et al., 2022). An ABC-SMC^2 method consists of three components: an outer SMC for parameter sampling, an inner ABC-SMC for updating the particle weights of the outer SMC, and an ABC-PMCMC incorporating the inner ABC-SMC to stochastically update the outer SMC particles to avoid particle weight degeneracy. Compared to simply implementing an ABC-PMCMC, the ABC-SMC^2 method starts with multiple starting values that evolve and interact with others in the outer SMC. It thus mitigates the issue of getting stuck in a local mode, and the particles can better represent the posterior of the parameters.

1.4 Summary and Contribution

We have presented four popular SMC methods (the bootstrap filter, the auxiliary particle filter, Storvik's filter, and Liu and West's filter) in the aforementioned sections. These algorithms are relatively easy to implement and are still popular to use in recent research. Based on these, several variants of SMC have also been proposed such as the lookahead SMC proposed by Lin et al. (2013) which considers future information when evaluating particle weights, the annealed SMC sampler Del Moral et al. (2006); Wang et al. (2020), and the conditional SMC (Andrieu et al., 2010; Lindsten et al., 2014) which fixed an input particle throughout the propagation procedure. In Chapter 2, we propose a new variant of SMC for the problem of sampling protein structures in structural biology, which draws multiple descendants for each particle during propagation and resamples from them to maintain the desired particle size.

Some SMC-based methods including the PMCMC-class methods and the SMC^2-class methods have also been proposed that adopt the SMC framework as a functional part of larger algorithms used for parameter inference. PMCMC was originally designed for parameter learning, which implements an SMC method to approximate the particle weights in SSMs within an MCMC algorithm. Moreover, it allows us to solve parameter learning for HMMs/SSMs where the likelihood is intractable in ABC and is, to our best knowledge, the only class of approaches to this problem (Vankov et al., 2019). In Chapter 3, we present a project where we propose a particle Gibbs sampler, a special version of PMCMC, adopting an ABC-based conditional APF for the stochastic volatility model with intractable likelihood in financial time series.

As an MCMC-based method, PMCMC adopts the features of standard MCMC such as the Metropolis-Hastings algorithm (Metropolis et al., 1953; Hastings, 1970): theoretically, it will eventually converge and generate samples from the target distribution, but it can easily get stuck in some local modes and lead to extremely slow convergence. This issue usually becomes severe in ABC because the target distribution can have very restricted supports or high-density regions. To handle this problem, one may consider developing an SMC^2 algorithm that embeds PMCMC moves (Chopin et al., 2013; Kerama et al., 2022). In Chapter 4, we present a project where we implement an SMC^2 algorithm for inference on a compartmental model with intractable transmission probabilities in epidemiological studies.

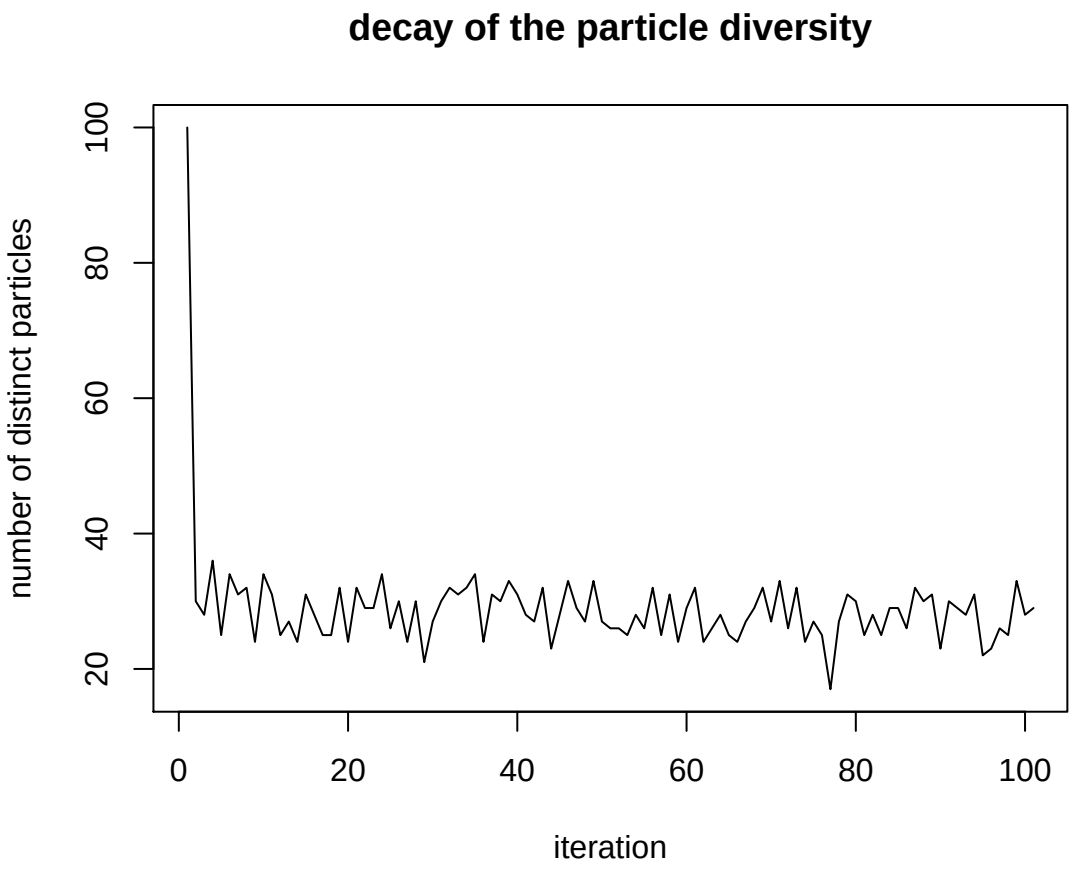


Figure 1.1: The plot of the number of the distinct particles over iteration times through the bootstrap filter. For simplicity, the state space of the state x_t is $\{1, 2, \ldots, 100\}$ with uniform prior and the weight of $x_{1:T}$ is proportional to $\prod_{t=1}^{T} \exp(x_t/10)$. With 100 different starting values as the initial value for each particle, the diversity quickly drops to around 30.

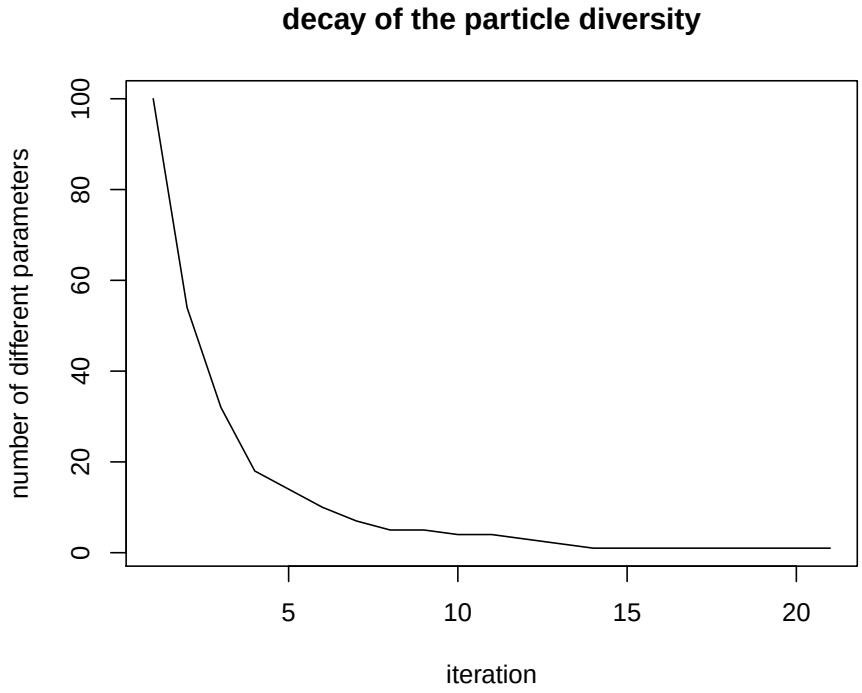


Figure 1.2: The plot of the number of different parameters evolving along with the hidden states in the bootstrap filter. $\{x_0^{(n)} = 0\}_{n=1}^{100}$ each with $w_0^{(n)} = n$, we assume $x_{t+1} \mid x_t \sim N(x_t, 1)$, $y_t \mid x_t, \omega \sim N(x_t, \omega^2)$ and $y_t = 0$ for all t. The number of different parameters starts from 100, drops below 10 after several iterations and ends up with 1.

Chapter 2

Estimating Boltzmann Averages for Protein Structural Quantities Using Sequential Monte Carlo

2.1 Introduction

Sequential Monte Carlo (SMC) methods, also known as particle filters, are simulation-based Monte Carlo algorithms for sampling from a target distribution. SMC originated from on-line inference problems in dynamic systems, where observations arrive sequentially and interest lies in the posterior distribution of hidden state variables (Liu and Chen, 1998). Subsequent developments include the Rao-Blackwellised particle filter and its extensions (Casella and Robert, 1996; Chen and Liu, 2000; Andrieu and Doucet, 2002; Chen et al., 2010; Johansen et al., 2012) and the class of particle Markov Chain Monte Carlo algorithms (Andrieu et al., 2010; Kantas et al., 2015; Chopin and Singh, 2015). These methods specialize in handling dynamic systems and their structure of hidden states. SMC has also been adapted as a useful approach for sampling from general high-dimensional probability distributions (Liu, 2001; Del Moral et al., 2006; Wang et al., 2020); this is the setting considered in this paper.

We begin with a review of the relevant SMC concepts for general sampling problems following Doucet et al. (2001). Assume we have a vector of random variables $(\mathbf{x}_0, \ldots, \mathbf{x}_T)$, denoted by $\mathbf{x}_{0:T}$, with continuous support $\mathcal{X}^{T+1}$, and we wish to draw samples from the target distribution $p(\mathbf{x}_{0:T})$. Let $f_T : \mathcal{X}^{T+1} \to \mathbb{R}^{n_{f_T}}$ denote a square integrable function of

interest, then its expectation with respect to $p(\mathbf{x}_{0:T})$ is given by

$$E_p\left[f_T\left(\mathbf{x}_{0:T}\right)\right] = \int f_T\left(\mathbf{x}_{0:T}\right) p\left(\mathbf{x}_{0:T}\right) d\mathbf{x}_{0:T}. \tag{2.1}$$

Since this integration is usually analytically intractable, the goal of SMC is to produce a set of particles $\{(\mathbf{x}_{0:T}^{(n)}, w(\mathbf{x}_{0:T}^{(n)}))\}_{n=1}^N$ that is proper with respect to $p(\mathbf{x}_{0:T})$ (Liu and Chen, 1998; Liu, 2001; Liu et al., 2001), i.e., $E[f_T(\mathbf{x}_{0:T}^{(n)})w(\mathbf{x}_{0:T}^{(n)})]$ and $E[w(\mathbf{x}_{0:T}^{(n)})]$ do not depend on n and satisfy

$$\frac{E[f_T(\mathbf{x}_{0:T}^{(n)})w(\mathbf{x}_{0:T}^{(n)})]}{E[w(\mathbf{x}_{0:T}^{(n)})]} = E_p\left[f_T(\mathbf{x}_{0:T})\right], \tag{2.2}$$

then an estimate to (2.1) is given by $\widehat{E}_p\left[f_T\left(\mathbf{x}_{0:T}\right)\right] = \frac{\sum_{n=1}^N f_T(\mathbf{x}_{0:T}^{(n)})w(\mathbf{x}_{0:T}^{(n)})}{\sum_{n=1}^N w(\mathbf{x}_{0:T}^{(n)})}$. Often, $p(\mathbf{x}_{0:T})$ does not adopt a form from which we can efficiently sample in practice (e.g., Jacquier et al. (2002); Carvalho et al. (2010)). In this case, a sequence of importance distributions $\eta(\mathbf{x}_0), \eta(\mathbf{x}_1 \mid \mathbf{x}_0), \ldots, \eta(\mathbf{x}_T \mid \mathbf{x}_{0:T-1})$ and a set of extended auxiliary distributions $\{p_t(\mathbf{x}_{0:t})\}_{t=0}^T$, with $p_t(\mathbf{x}_{0:t})$ defined on $\mathcal{X}^{t+1}$ and $p_T(\mathbf{x}_{0:T}) = p(\mathbf{x}_{0:T})$, can be introduced (Liu, 2001). To then construct $\{\mathbf{x}_{0:T}^{(n)}\}_{n=1}^N$, SMC generates particles according to the auxiliary distributions via a sequence of propagation and resampling steps, e.g., via the sequential importance sampling with resampling (SISR) algorithm (e.g., Liu and Chen, 1995, 1998, see Section S1 in the Supplementary Materials for a summary).

The motivating application of the SMC method proposed in this paper is for sampling 3-D structures of proteins. In this context, $\mathbf{x}_{0:T}$ is a vector of dihedral angles (with each angle defined on the space [-180°, 180°)) which corresponds to a *conformation*, i.e., an arrangement of the protein's atoms in 3-D space. The sampling target is the Boltzmann distribution (Boltzmann, 1868; Landau and Lifshitz, 2013), defined by

$$p(\mathbf{x}_{0:T}) \propto \exp\left\{-H(\mathbf{x}_{0:T})/\lambda\right\} \tag{2.3}$$

where H is a given energy function, and λ is the effective temperature which can be taken to be 1 by appropriately scaling H. Based on a properly weighted Monte Carlo sample, we can estimate $E_p[f(\mathbf{x}_{0:T})]$ with respect to (2.3), where f is a quantity of interest (i.e., a given function evaluated on conformations $\mathbf{x}_{0:T}$); these are known as *Boltzmann averages* that represent the conformational equilibrium of the quantity f (Zhou and Berne, 1997). In practice, much of the space of (2.3) has density zero due to atomic and geometric constraints, so sampling from the Boltzmann distribution for protein structures is a challenging task. While related to the famous *protein folding problem*, the latter focuses on finding the most likely conformation, i.e., $\arg\max p(\mathbf{x}_{0:T})$ (Anfinsen, 1973; Onuchic et al.,

1997), and is thus distinct from our goal. The problem of estimating Boltzmann averages is traditionally handled by using molecular dynamics (MD) or Markov chain Monte Carlo (MCMC) simulations (Adcock and McCammon, 2006). However, these methods tend to be time-consuming and suffer from being trapped in local modes; an effective SMC method to circumvent these difficulties has only been proposed for sampling simplified discrete representations of protein structures (Zhang et al., 2007).

SMC is an intuitive strategy for sampling from (2.3) since proteins are by nature a sequence of amino acids (see Section 2.4 for scientific background). However, a simple SISR algorithm suffers from weight degeneracy, i.e., many of the intermediate particle weights decay to zero as a result of the highly constrained support of (2.3) (Wong et al., 2018). Some general techniques have been developed for handling weight degeneracy. Different resampling schemes have been proposed to improve the performance of SISR (Gordon et al., 1993; Kitagawa, 1996; Liu and Chen, 1998; Li et al., 2022), but resampling alone cannot thoroughly solve weight degeneracy for all situations. The lookahead strategies considered in Lin et al. (2013) incorporate information from future steps into current particles to help reduce weight degeneracy, but our experiments (Section 2.5) show only limited success for protein sampling. Annealed importance sampling can be adopted within an SMC algorithm to temper the auxiliary distributions (Neal, 2001; Del Moral et al., 2006; Dai et al., 2022), but tempering is ineffective when many of the particle weights are exactly zero. Fearnhead and Clifford (2003) proposed to exhaustively explore the space by generating multiple descendants for each particle and then resampling; this strategy can help circumvent weight degeneracy but is only applicable to finite spaces.

In this paper, we adopt the idea of generating multiple descendants and resampling from Fearnhead and Clifford (2003), and extend it to continuous spaces as required for sampling protein conformations from (2.3). The proper weighting condition is maintained, so that we may obtain consistent estimates of Boltzmann averages. While motivated by the sampling problem in the protein context, the proposed SMC strategy is generally applicable for sampling from multivariate continuous distributions to compute Monte Carlo integrals. It may be especially effective when the support of the target distribution is highly constrained and weight degeneracy renders the SISR algorithm inapplicable.

The remainder of the paper is laid out as follows. In Section 2.3, we present the construction of our SMC method. In Section 2.4, we introduce the scientific background of proteins and the quantities of interest that may be computed from protein structures. In Section 2.5, we implement a simulation study that illustrates the advantages of the proposed method compared to existing ones for sampling protein structures. In Section 2.6, we apply the proposed SMC method to estimate the number of atomic contacts for a key segment of the SARS-CoV-2 viral spike protein. In Section 2.7, we briefly summarize

the paper and its contributions and discuss some potential future directions. Proofs are provided in the Supplementary Materials.

2.2 Motivating Application: Estimating Protein Structural

In this paper, we focus on the 3-D structures of protein segments (also known as loops) due to the important roles they can have in biological function, e.g., the highly dynamic region of the coronavirus spike protein binding with human cells (see the real data example in Section 2.6). The structure prediction problem for segments, with the rest of the protein held fixed, has thus attracted much attention. The basic idea of computational structure prediction is to find the conformation with the lowest potential energy; this idea is based on the energy-landscape theory of protein folding (Onuchic et al., 1997). Prediction methods have included sequential sampling approaches (Tang et al., 2014; Wong et al., 2018), robotics-inspired protocols (Stein and Kortemme, 2013), and hybrid approaches that leverage databases of known loops (Marks et al., 2017). These methods focus on predicting a single "correct" conformation with the lowest potential energy. However, for exploring a real protein's range of dynamic movement and estimating Boltzmann averages (according to a given energy function), new sampling approaches are needed.

In analyses of protein structures, various quantities or summary statistics are computed from conformations, depending on the intended application. The C_α backbone atoms often have an important role, e.g., contact prediction considers Euclidean distances among all pairs of C_α's (Di Lena et al., 2012; Zheng et al., 2019); the number of other C_α atoms within a sphere centered at an amino acid's C_α is counted for solvent surface estimation (Simons et al., 1997); the C_α atom is often chosen as a substitute for the entire amino acid in coarse-grain protein representations, to improve the computational efficiency of structure and function prediction (Roy et al., 2010). Statistics involving *atomic contacts*, i.e., the number of non-bonded atoms within a given radius of a selected atom, are also important: Karlin et al. (1999) investigated atom density in terms of the average number of neighboring C, O, and N atoms; Pintar et al. (2002) introduced the CX algorithm to identify protruding atoms in terms of the number of non-hydrogen atoms within a given radius; Mihel et al. (2008) developed a software tool that integrates several algorithms, including CX, to compute protein quantities for protein interaction; Ligeti et al. (2017) developed the Protein Core Workbench (PCW) server for explicit visualization of atomic contacts. Protruding atoms, i.e., those with fewer atomic contacts, tend to be on the surface of the protein and more involved in binding activities (Pintar et al., 2002).

Therefore, to illustrate our methodology we consider similar statistics that summarize aspects of protein segment conformations: (i) distances between pairs of C_α atoms, as used in contact prediction; (ii) number of atomic contacts of C_α atoms, within a radius of 7 Å.

2.3 Methodology

2.3.1 Review of Fearnhead and Clifford's SMC method

To introduce the key ideas, we review the relevant details of the SMC method proposed by Fearnhead and Clifford (2003), which was developed from the (partial) rejection control method (Liu et al., 1998, 2001). Consider a hidden Markov model setup with hidden process $\{\mathbf{x}_t\}_{t=0}^T$, each with finite state space $\mathcal{X}$ such that $|\mathcal{X}| = M < \infty$, observations $\{y_t\}_{t=1}^T$, distribution of the initial state $p(\mathbf{x}_0)$, transition probability $q(\mathbf{x}_t \mid \mathbf{x}_{t-1})$ and observation probability $l(y_t \mid \mathbf{x}_t)$ for $0 < t \leq T$. Their goal was to sample from the posterior

$$p(\mathbf{x}_{0:T} \mid y_{1:T}) \propto p(\mathbf{x}_0) \prod_{t=1}^{T} q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) l(y_t \mid \mathbf{x}_t).$$

Given a set of weighted particles $\{\mathbf{x}_{0:t-1}^{(n)}\}_{n=1}^N$ up to index $t-1$, each $\mathbf{x}_{0:t-1}^{(n)}$ is used to produce M distinct descendants, denoted by $\mathbf{x}_t^{(n,m)}$, one for each value of $\mathcal{X}$. The weight of the propagated particle $\mathbf{x}_{0:t}^{(n,m)} = (\mathbf{x}_{0:t-1}^{(n)}, \mathbf{x}_t^{(n,m)})$ is given by

$$w(\mathbf{x}_{0:t}^{(n,m)}) = w(\mathbf{x}_{0:t-1}^{(n)}) q(\mathbf{x}_t^{(n,m)} \mid \mathbf{x}_{0:t-1}^{(n)}) l(y_t \mid \mathbf{x}_t^{(n,m)}). \tag{2.4}$$

To resample N particles from the NM candidates, the constant c_t in

$$\sum_{n=1}^{N} \sum_{m=1}^{M} \min(c_t w(\mathbf{x}_{0:t}^{(n,m)}), 1) = N \tag{2.5}$$

is calculated. Let L be the number of particles with weights greater than or equal to $1/c_t$; these L particles are all preserved, and stratified sampling (Carpenter et al., 1999) is used to resample another $N - L$ particles without replacement. Fearnhead and Clifford (2003) showed that this downsampling scheme minimizes

$$E\left[\sum_{n=1}^{N} \sum_{m=1}^{M} (Q(\mathbf{x}_{0:t}^{(n,m)}) - \gamma_t^{(n,m)})^2\right]$$

where $\gamma_t^{(n,m)}$ denotes the weight with respect to p_t and $Q(\mathbf{x}_{0:t}^{(n,m)})$ is the stochastic weight of the sampled particle, and thus is optimal among downsampling schemes. For a chosen particle size N, the computational complexity of the algorithm is $O(NM)$ since the evaluation of the incremental weights in (2.4) tends to be the most expensive part of the computation in practice; in contrast, the cost of SISR is $O(N)$ in this regard.

As a generalization of these key ideas, we shall, in the following context, define *upsampling* to be a propagation step that samples $M \geq 1$ descendants from each of the N existing particles, and *downsampling* to be the step of resampling N particles from the NM total descendants. An *upsampling-downsampling framework* shall refer to an SMC algorithm that combines these upsampling and downsampling features.

2.3.2 A general upsampling-downsampling SMC framework

Our goal is to sample from a general multivariate distribution $p(\mathbf{x}_{0:T})$ where $\mathbf{x}_t$ is a random vector with continuous support, with the help of an upsampling-downsampling SMC framework (or *UDSMC* for short). In this situation, it is impossible to explore every value of $\mathcal{X}$, so we use importance distributions $\eta(\mathbf{x}_0), \eta(\mathbf{x}_1 \mid \mathbf{x}_0), \ldots, \eta(\mathbf{x}_T \mid \mathbf{x}_{0:T-1})$ to facilitate sampling.

Assume for a chosen particle size N, upsample size M, and $t > 0$, we have a set of generated particles $\{\mathbf{x}_{0:t-1}^{(n)}\}_{n=1}^N$ with weights $\{w(\mathbf{x}_{0:t-1}^{(n)})\}_{n=1}^N$. For each $\mathbf{x}_{0:t-1}^{(n)}$, we sample M descendants $\{\mathbf{x}_t^{(n,m)}\}_{m=1}^M$ from $\eta(\mathbf{x}_t \mid \mathbf{x}_{0:t-1}^{(n)})$ and define $\mathbf{x}_{0:t}^{(n,m)} = (\mathbf{x}_{0:t-1}^{(n)}, \mathbf{x}_t^{(n,m)})$ with

$$w(\mathbf{x}_{0:t}^{(n,m)}) = \frac{w(\mathbf{x}_{0:t-1}^{(n)}) p_t(\mathbf{x}_{0:t}^{(n,m)})}{p_{t-1}(\mathbf{x}_{0:t-1}^{(n)}) \eta(\mathbf{x}_t^{(n,m)} \mid \mathbf{x}_{0:t-1}^{(n)})}. \tag{2.6}$$

These NM propagated particles with the upsampled weights in (2.6) are then carried to the downsampling step to resample N particles.

The downsampling step resamples N particles from the set $\{\mathbf{x}_{0:t}^{(m,n)}, n = 1, \ldots, N \text{ and } m = 1, \ldots, M\}$; the resampled particles are then denoted by $\{\mathbf{x}_{0:t}^{(n)}\}_{n=1}^N$. There are two possible cases after each upsampling step: (i) at least N of $w(\mathbf{x}_{0:t}^{(n,m)})$'s are positive and (ii) less than N of $w(\mathbf{x}_{0:t}^{(n,m)})$'s are positive. (Note that Fearnhead and Clifford (2003) did not need to consider case (ii) as the Gaussian observation likelihood in their application always produces positive weights after upsampling.) Handling case (ii) ensures a valid algorithm when weight degeneracy is very severe.

For case (i), the downsampling step follows the method of Fearnhead and Clifford (2003). After the t-th upsampling step, the threshold c_t in the equation

$$\sum_{n=1}^{N}\sum_{m=1}^{M}\min\left\{c_t w(\mathbf{x}_{0:t}^{(n,m)}), 1\right\} = N$$

is calculated. Let L be the number of particles whose weights are greater than or equal to $1/c_t$; these are preserved together with the weights $w(\mathbf{x}_{0:t}^{(n,m)})$. $N - L$ particles are sampled without replacement from the remaining $NM - L$ particles proportional to their weights $w(\mathbf{x}_{0:t}^{(n,m)})$ and are assigned new weights $1/c_t$. Therefore, after the downsampling step, each $\mathbf{x}_{0:t}^{(n,m)}$ has a stochastic downsampled weight $Q(\mathbf{x}_{0:t}^{(n,m)})$ as

$$Q(\mathbf{x}_{0:t}^{(n,m)}) = \begin{cases} w(\mathbf{x}_{0:t}^{(n,m)})/q(\mathbf{x}_{0:t}^{(n,m)}) & \text{with probability } q(\mathbf{x}_{0:t}^{(n,m)}) \\ \\ 0 & \text{otherwise} \end{cases} \tag{2.7}$$

where $q(\mathbf{x}_{0:t}^{(n,m)}) = \min\{c_t w(\mathbf{x}_{0:t}^{(n,m)}), 1\}$. This resampling scheme ensures that the downsampled weights $Q(\mathbf{x}_{0:t}^{(n,m)})$ are proper with respect to p_t. This resampling scheme is optimal in terms of minimizing the expected squared error loss when resampling N distinct particles from the given collection of NM upsampled particles. These results are formalized in Propositions 1 and 2 below. The N downsampled particles are then the collection of those with realizations $Q(\mathbf{x}_{0:t}^{(n,m)}) > 0$, which we set to be $\{\mathbf{x}_{0:t}^{(n)}\}_{n=1}^{N}$ with corresponding weights $\{w(\mathbf{x}_{0:t}^{(n)})\}_{n=1}^{N}$.

Proposition 1. *For $t \in \{0, \ldots, T\}$ and the auxiliary distribution p_t, suppose there are at least N positive upsampled weights for all $s \in \{0, \ldots, t\}$. Let $\mathbf{x}_{0:t}$ be any particle from the space $\mathcal{X}^{t+1}$, then its downsampled particles $\{(\mathbf{x}_{0:t}^{(n,m)}, Q(\mathbf{x}_{0:t}^{(n,m)})); n = 1, \ldots, N, m = 1, \ldots, M\}$ given in Equation (2.7) are proper with respect to p_t.*

Proof. See Section S2 in the Supplementary Materials. □

Proposition 2. *For $t \in \{0, \ldots, T\}$ and upsampled particles $\{\mathbf{x}_{0:t}^{(n,m)}; n = 1, \ldots, N, m = 1, \ldots, M\}$, suppose there are at least N positive upsampled weights at step t. When each individual downsampled weight $Q(\mathbf{x}_{0:t}^{(n,m)})$ is assigned by Equation (2.7) with the constraint that no more than N elements of $\{Q(\mathbf{x}_{0:t}^{(n,m)}); n = 1, \ldots, N, m = 1, \ldots, M\}$ are positive, then the downsampled weights minimize the conditional expected squared error loss*

$$E\left[\sum_{n=1}^{N}\sum_{m=1}^{M}(Q(\mathbf{x}_{0:t}^{(n,m)}) - \gamma_t^{(n,m)})^2 \,\middle|\, \{\mathbf{x}_{0:t}^{(n,m)}; n = 1, \ldots, N, m = 1, \ldots, M\}\right]$$

Algorithm 1: UDSMC

Require: particle size N, upsample size M;

Initial upsampling: Sample $\{\mathbf{x}_0^{(n,m)}, n = 1, \ldots, N \text{ and } m = 1, \ldots, M\}$ from $\eta(\mathbf{x}_0)$, each with the weight $w(\mathbf{x}_0^{(n,m)}) = p_0(\mathbf{x}_0^{(n,m)})/\eta(\mathbf{x}_0^{(n,m)})$;

Initial downsampling: Resample N particles, denoted by $\{\mathbf{x}_0^{(n)}\}_{n=1}^N$, from $\{\mathbf{x}_0^{(n,m)}, n = 1, \ldots, N \text{ and } m = 1, \ldots M\}$ with weights $\{w(\mathbf{x}_0^{(n)})\}_{n=1}^N$ using the proposed scheme;

for $t = 1, \ldots, T$ **do**

Upsampling: Sample $\{\mathbf{x}_t^{(n,m)}\}_{m=1}^M$ from $\eta(\mathbf{x}_t \mid \mathbf{x}_{0:t-1}^{(n)})$ and set $\mathbf{x}_{0:t}^{(n,m)} = (\mathbf{x}_{0:t-1}^{(n)}, \mathbf{x}_t^{(n,m)})$ for each n, each with a weight $w(\mathbf{x}_{0:t}^{(n,m)}) = w(\mathbf{x}_{0:t-1}^{(n)}) p_t(\mathbf{x}_{0:t}^{(n,m)})/(p_{t-1}(\mathbf{x}_{0:t-1}^{(n)})\eta(\mathbf{x}_t^{(n,m)} \mid \mathbf{x}_{0:t-1}^{(n)}))$;

Downsampling: Resample N particles, denoted by $\{\mathbf{x}_{0:t}^{(n)}\}_{n=1}^N$, from $\{\mathbf{x}_{0:t}^{(n,m)}, n = 1, \ldots, N \text{ and } m = 1, \ldots M\}$ with weights $\{w(\mathbf{x}_{0:t}^{(n)})\}_{n=1}^N$ using the proposed scheme;

where $\gamma_t^{(n,m)} = \left(p_t(\mathbf{x}_{0:t}^{(n,m)})/\eta(\mathbf{x}_{0:t}^{(n,m)})\right) / \sum_{n=1}^N \sum_{m=1}^M \left(p_t(\mathbf{x}_{0:t}^{(n,m)})/\eta(\mathbf{x}_{0:t}^{(n,m)})\right)$.

Proof. See Section S3 in the Supplementary Materials. □

For case (ii), we have less than N particles with positive weights so cannot preserve N distinct particles via downsampling; instead, resampling N particles with replacement is needed, similar to the resampling step in SISR. The rationale for resampling in this case is to effectively duplicate particles with high weights, thereby retaining more potentially feasible particles as starting points for the following propagation step. We use multinomial resampling for simplicity, which mimics the resampling scheme of the bootstrap filter (Gordon et al., 1993), but other schemes can also be used. The key difference between case (i) and (ii) is that there are duplicates of particles with equal weights after downsampling for case (ii).

Finally to initialize the algorithm, NM realizations of $\mathbf{x}_0$, i.e. $\{\mathbf{x}_0^{(n,m)}, n = 1, \ldots N \text{ and } m = 1, \ldots, M\}$, are first sampled from $\eta(\mathbf{x}_0)$ (each with importance weight $p_0(\mathbf{x}_0)/\eta(\mathbf{x}_0)$), followed by a downsampling step to obtain N properly weighted particles representing $p_0(\mathbf{x}_0)$. Algorithm 1 summarizes UDSMC.

Analogously to that described in Section 2.3.1, UDSMC has computational complexity $O(NM)$ due to the evaluation of the incremental weights in (2.6). Thus, our method is

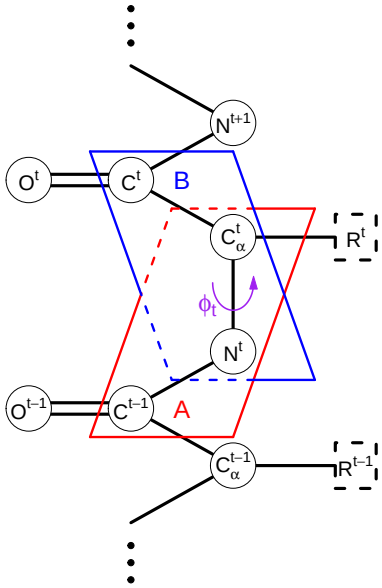
best suited for situations where running SISR with a very large particle size (e.g., larger than NM in Algorithm 1) does not produce satisfactory results; the subsequent protein application is one such situation. Moreover, the performance of Algorithm 1 will be influenced by the choice of the upsample size M. For a finite space $\mathcal{X}$, a natural choice for M can be to take $M = |\mathcal{X}|$; however, there may not be an intuitive choice for M for a continuous space $\mathcal{X}$. Thus to choose a reasonable value of M in a given application, we may run preliminary experiments using different values of M (with a fixed computational budget, i.e., holding MN constant), as we subsequently demonstrate.

2.4 Estimating Protein Structural Quantities

Proteins have a crucial role in carrying out biological processes and their functions are dependent on their 3-D structures. A protein is composed of a sequence of amino acids, by which its 3-D structure is essentially determined (Anfinsen, 1973). However, it is also well-known that protein structures are not static; some dynamic movement is observed (Fraser et al., 2009, 2011; Bu and Callaway, 2011) and the relative probability of different conformations can be characterized by the Boltzmann distribution defined in (2.3), i.e., conformations with lower energy are more favorable. We focus on the 3-D structures of key protein segments due to their important biological functions, e.g., the highly dynamic region of the coronavirus spike protein that binds with human cells (Lan et al., 2020). While the Protein Data Bank (PDB) (Berman et al., 2000) is the source of known protein structures obtained from laboratory work, these should be considered as only static snapshots of a given protein. To study the dynamic movement of a protein and estimate Boltzmann averages, efficient computational approaches are needed for sampling conformations, which motivates our current work.

The amino acid indexed by t in the sequence is composed of four *backbone* atoms (the nitrogen atom denoted by N^t, the carbon atoms denoted by C_α^t and C^t, and the oxygen atom denoted by O^t) and a *side chain* denoted by R^t. Figure 2.1 illustrates how these backbone atoms and side chains are connected, with the solid lines indicating bonds. Successive amino acids are also connected, e.g., the bond between C^t and N^{t+1} connects the amino acids indexed by t and $t+1$ in Figure 2.1. The 3-D coordinates of the backbone atoms $a_t = (\mathrm{C}^t, \mathrm{O}^t, \mathrm{N}^{t+1}, \mathrm{C}_\alpha^{t+1})$ can be equivalently specified using a vector of dihedral angles $(\phi_t, \psi_t, \omega_t)$, where ϕ_t represents the dihedral angle of $\mathrm{C}^{t-1} - \mathrm{N}^t - \mathrm{C}_\alpha^t$ and $\mathrm{N}^t - \mathrm{C}_\alpha^t - \mathrm{C}^t$; ψ_t represents the dihedral angle of $\mathrm{N}^t - \mathrm{C}_\alpha^t - \mathrm{C}^t$ and $\mathrm{C}_\alpha^t - \mathrm{C}^t - \mathrm{N}^{t+1}$; ω_t represents the dihedral angle of $\mathrm{C}_\alpha^t - \mathrm{C}^t - \mathrm{N}^{t+1}$ and $\mathrm{C}^t - \mathrm{N}^{t+1} - \mathrm{C}_\alpha^{t+1}$. Figure 2.1 demonstrates an example of how ϕ_t is determined, and ψ_t is determined in a similar way by shifting the planes up by one

Figure 2.1: Illustration of connections and dihedral angles in a protein segment. Solid lines in the figure indicate bonds (connections). Within the amino acid indexed by t, N^t, C_α^t, C^t and O^t are successively connected, and the side chain R^t (that distinguishes different amino acid types) is connected to C_α^t; the double lines between C^t and O^t indicate a double-bond connection. Successive amino acids are connected via a C-N bond, e.g., amino acids indexed by t and $t+1$ are connected with the bond between C^t and N^{t+1}. To define the dihedral angle ϕ_t, consider the two planes marked in the figure: Plane A (red) consists of atoms C^{t-1}, N^t and C_α^t; Plane B (blue) consists of atoms N^t, C_α^t and C^t. Then ϕ_t is the angle between Plane A and Plane B when looking down the $\mathrm{N}-\mathrm{C}_\alpha$ bond.



atom. Since bond lengths and bond angles are essentially fixed, ϕ_t governs the distance between C^{t-1} and C^t; ψ_t governs the distance between N^t and N^{t+1} and also determines the coordinates of O^t. The bond connecting C^t and N^{t+1} is nearly non-rotatable, and thus the dihedral angle ω_t that governs the distance between C_α^t and C_α^{t+1} is usually close to 180° (Esposito et al., 2005). In contrast, ϕ_t and ψ_t can take a wide range of values over the continuous interval [-180°, 180°).

Within a given protein, suppose that the segment of interest consists of the amino acids indexed from j to k, denoted by $(a_j, \ldots, a_k)$. The backbone structure of the segment can then be parameterized by $\mathbf{x}_{0:T} = (\mathbf{x}_0, \ldots, \mathbf{x}_T)$ with $T = k - j$, where $\mathbf{x}_t$, $t \in \{0, \ldots, T\}$, represents the three dihedral angles for the amino acid a_{j+t}, i.e., $\mathbf{x}_t = (\phi_{j+t}, \psi_{j+t}, \omega_{j+t})$. To focus on the properties of the segment, the rest of the protein is held fixed (e.g., at a static snapshot obtained from the PDB) while sampling conformations for the backbone of $(a_j, \ldots, a_k)$. For simplicity, the side chains R^t are not considered in this work.

The energy function H in (2.3) consists of two components: following Wong et al. (2018), for a given backbone segment $\mathbf{x}_{0:T}$, the energy of atomic interactions $H_a(\mathbf{x}_{0:T})$ is based on pairwise atom distances (between atoms within the segment, and also atoms between the segment and the rest of the protein), and the energy of dihedral angles $H_\theta(\mathbf{x}_{0:T})$ is based on empirical distributions derived from historical protein data (Wong et al., 2017). The total energy of the segment is $H(\mathbf{x}_{0:T}) = H_a(\mathbf{x}_{0:T}) + H_\theta(\mathbf{x}_{0:T})$; detailed calculations for H_a and H_θ are presented in Section S4 of the Supplementary Materials. It should be noted that compared to H_θ, H_a is expensive to compute since all pairwise distances between atoms need to be calculated; moreover, if two atoms violate geometric constraints (e.g., they are too close together in 3-D space), then the conformation will have $H_a = +\infty$ which implies a zero weight. This property of H_a leads to a highly constrained support for (2.3), which poses a challenge for sampling methods.

In protein structure analysis, summary statistics $f(\mathbf{x}_{0:T})$ can be computed; we call these *protein structural quantities*. The C_α backbone atoms are often treated as the representative of each amino acid for such computations (Simons et al., 1997; Di Lena et al., 2012). Summary statistics involving *atomic contacts*, i.e., the number of other atoms within a given radius of a selected atom, help determine whether the selected atom is close to the protein's surface (Karlin et al., 1999; Pintar et al., 2002). Within a given protein segment, C_α atoms with fewer atomic contacts tend to be on the surface of the protein and therefore more likely to interact with other molecules. To illustrate our methodology, we consider using the samples from (2.3) to estimate $E_p[f(\mathbf{x}_{0:T})]$ for the atomic contacts of C_α atoms within a radius of 7 Å.

2.5 Simulation study

As a simulation study, we consider the length 8 segment from a_{284} to a_{291} of the protein with PDB ID *1ds1A*. The protein structural quantity of interest is the atomic contacts for each C_α in the segment, denoted by $n(C_\alpha^{285}), \ldots, n(C_\alpha^{292})$. This protein segment is of sufficient difficulty (the dimension of the Boltzmann distribution is 24 in this example) to showcase a comparison of different methods, while at the same time is simple enough such that the ground truth of the Boltzmann averages can be obtained via brute force.

To obtain the ground truth for this example, the brute force approach we used is as follows. We sequentially draw samples from the importance distributions $\eta(\mathbf{x}_0), \eta(\mathbf{x}_1 \mid \mathbf{x}_0), \ldots, \eta(\mathbf{x}_7 \mid \mathbf{x}_{0:6})$, defined by

$$\begin{aligned} \eta(\mathbf{x}_0) &\propto \exp\left(-H_\theta(\mathbf{x}_0)\right) \\ \eta(\mathbf{x}_t \mid \mathbf{x}_{0:t-1}) = \eta(\mathbf{x}_t) &\propto \exp\left(-H_\theta(\mathbf{x}_t)\right) \quad t = 1, \ldots 7, \end{aligned} \tag{2.8}$$

since H_θ is constructed to be independent for each amino acid position (see Section S4 of the Supplementary Materials). Note that any importance distribution could be used here; H_θ is chosen for this purpose as it is easy to sample from the empirical distribution of dihedral angles and is more efficient than drawing $\mathbf{x}_t$ uniformly. Given that a partial sequence from $\mathbf{x}_0$ to $\mathbf{x}_s$ with $s \in \{0, \ldots, 6\}$ has been generated, we draw descendants from $\eta(\mathbf{x}_{s+1})$ until a sample with positive importance weight is obtained, i.e., $H(\mathbf{x}_{0:s+1}) < +\infty$; if it is not possible to obtain such a sample, we discard the partial sequence and start over. After successfully simulating a complete conformation, the total energy $H(\mathbf{x}_{0:7})$ is evaluated to obtain its importance weight.

Table 2.1 displays the ground truth for the Boltzmann averages of the eight quantities of interest, as approximated from a total of 10^8 samples (with positive importance weights). To verify that 10^8 samples provide a good proxy for the ground truth here, we randomly divided them into two subgroups of 5×10^7 samples, repeating 50 times to form a total of 100 groups of 5×10^7 samples. The same quantities were estimated from each of the 100 groups. For each quantity, the standard deviations of these estimates were less than 0.4% of the Boltzmann average computed from the full (10^8) sample, which indicates the stability desired. Overall, this computation of the ground truth was expensive: it required one week of compute time on a cluster of 300 Intel Xeon CPU cores.

Having established a proxy for the ground truth, we proceed to compare the performance of three different SMC methods in this section. For a fair comparison, the importance distributions for the propagation step are chosen to be the same for each SMC

Table 2.1: Boltzmann averages of the eight quantities of interest. The 10^8 samples obtained using our brute force approach were used as a proxy for the ground truth.

$n(C_\alpha^{285})$	$n(C_\alpha^{286})$	$n(C_\alpha^{287})$	$n(C_\alpha^{288})$	$n(C_\alpha^{289})$	$n(C_\alpha^{290})$	$n(C_\alpha^{291})$	$n(C_\alpha^{292})$
35.135	43.042	45.313	35.236	38.300	55.303	53.273	65.142

method, namely by adopting η as defined in (2.8). The first one is UDSMC. We implement UDSMC with an upsample size $M = 20$, based on the results of an experiment that assessed the Monte Carlo variance of different values of M with the overall computational budget fixed (see details in Section S5 of the Supplementary Materials).

The second one is the SISR algorithm, as summarized in Section S1 of the Supplementary Materials. SISR is commonly used and easy to implement, but it can encounter problems when weight degeneracy is severe and can fail to complete (i.e., all particles have zero weights before any complete conformations are simulated). Here, if SISR fails to complete, we simply restart the algorithm from the beginning.

The third one is SISR with the lookahead strategy, or simply *lookahead SISR*, as detailed in Section S6 of the Supplementary Materials. The lookahead strategy of Lin et al. (2013) utilizes "future" information for inference on the "current" states, and this idea was also adapted by Wong et al. (2018) for exploring protein structures in a finite space; here, we may apply this strategy in the context of SISR to alleviate the problem of severe weight degeneracy. After the t-th propagation step (i.e., producing $\mathbf{x}_{0:t}$), lookahead SISR implements an extra one-step exploration that evaluates multiple (L) "future" descendants (i.e., by generating $\{(\mathbf{x}_{0:t}, \mathbf{x}_{t+1}^l)\}_{l=1}^L$) and then implements a resampling step using the marginalized importance weights (i.e., $\sum_{l=1}^L w(\mathbf{x}_{0:t}, \mathbf{x}_{t+1}^l)$). In this way, the dead-end particles, i.e., where all L "future" descendants have zero weights, are discarded before resampling. We set $L = 20$ for lookahead SISR to mimic the upsample size of UDSMC in this simulation.

We run 100 repetitions of each method in the following and report the RMSEs of the estimates in Table 2.2. For UDSMC, we chose a range of N values from 5000 to 20,000; the computation speed is fast, e.g., $N = 10000$ and $M = 20$ had a time cost of about 200 seconds per repetition on a single core of a Xeon Gold 6244 3.6 GHz processor. To fairly compare the three methods, we assess their performance given a similar computational budget: given the N particles obtained from a resampling step, all three methods sample the same total number of descendants for the subsequent propagation step (i.e., by setting NM for UDSMC, N for SISR, and NL for lookahead SISR to be equal), since the evaluation of the energy function (specifically, H_a) is the computational bottleneck. For example, UDSMC with $N = 5000$ and $M = 20$ has similar computational cost as SISR with $N =$

$100,000$ and lookahead SISR with $N = 5000$ and $L = 20$.

Table 2.2: RMSEs of the eight quantities, based on 100 repetitions of UDSMC ($M = 20$), SISR and lookahead SISR ($L = 20$) for the protein segment $\mathbf{x}_{284:291}$. The corresponding rows of each method have a similar computational cost: e.g., UDSMC with $N = 5000$ and $M = 20$, SISR with $N = 100,000$, and lookahead SISR with $N = 5000$ and $L = 20$.

Method	$N =$	$n(C_\alpha^{285})$	$n(C_\alpha^{286})$	$n(C_\alpha^{287})$	$n(C_\alpha^{288})$	$n(C_\alpha^{289})$	$n(C_\alpha^{290})$	$n(C_\alpha^{291})$	$n(C_\alpha^{292})$
UDSMC	5000	0.612	1.576	1.556	1.372	2.794	2.701	1.539	0.576
($M = 20$)	10000	0.310	1.291	1.264	1.089	2.133	2.480	1.171	0.416
	20000	0.283	0.759	0.773	0.749	1.414	1.626	0.855	0.263
SISR	100000	4.825	8.382	9.638	10.058	15.342	15.236	10.019	3.347
	200000	2.988	8.020	7.995	9.115	13.843	14.489	9.546	3.248
	400000	1.528	7.773	7.213	8.701	11.944	13.372	9.019	2.404
Lookahead	5000	3.943	5.842	6.790	5.520	8.864	9.262	7.283	6.727
SISR	10000	1.190	2.879	3.858	3.375	5.482	6.070	4.408	1.088
($L = 20$)	20000	0.576	1.971	3.176	2.633	5.302	5.386	3.889	0.788

In Table 2.2, we observe that RMSEs decrease as the particle size increases for each SMC method; however, this pattern is not uniform across the amino acids in the segment. The RMSEs tend to be higher for the amino acids in the middle of the segment; this is sensible because amino acids further from the anchors tend to be less geometrically constrained by the rest of the protein. The RMSEs for amino acids in the middle of the segment also tend to decrease more slowly as particle size increases. Comparing the rows that have similar computational cost, we observe that SISR yields higher RMSEs compared to UDSMC and lookahead SISR (e.g., the RMSEs for SISR with $N = 200,000$ are substantively higher compared to UDSMC with $N = 10000$ and lookahead SISR with $N = 10000$), while UDSMC also outperforms lookahead SISR (with $> 50\%$ reduction in RMSE for every amino acid, compared to lookahead SISR for the same value of N). These simulation results indicate that UDSMC provides a more efficient approach to sampling conformations from the Boltzmann distribution compared to the two existing strategies. The relative inefficiency of SISR (even with a large N) makes sense: since when only a small fraction of the propagated particles have positive weights, there will be many duplicates among the N particles after a resampling step. This problem is exacerbated by successive iterations of propagation and resampling. For further discussion and a graphical example of how particle diversity decays, see Section S7 of the Supplementary Materials. Consequently, SISR can give poor estimates or even become entirely stuck in dead ends. Applying the lookahead strategy partially alleviates this problem, but is not nearly as effective as UDSMC for a given computational budget.

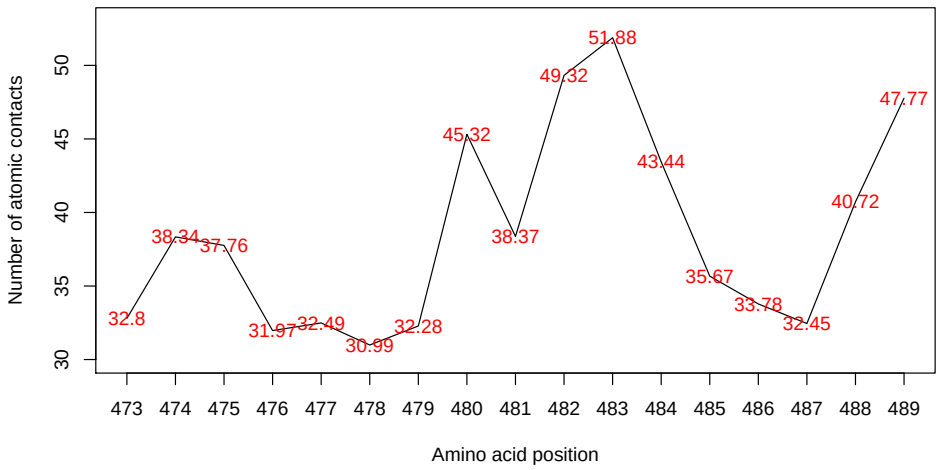
The protein segment used in this simulation study was short enough such that the ground truth could be obtained via brute force; in practice, target protein segments often have a much longer length (e.g., 12–20 amino acids long) so that the higher dimension of $\mathbf{x}$ poses a substantively more difficult sampling problem. The brute force approach used to facilitate benchmarking for this simulation study would no longer be applicable due to a prohibitive computational cost; e.g., it takes more than one hour to generate 10 valid samples for a typical length 18 segment (i.e., the length of our subsequent real data example) on a single core of a Xeon Gold 6244 3.6 GHz processor, and likely more than 10^8 samples would be needed to obtain stable estimates for the longer segment. Further, SISR will encounter such severe weight degeneracy that it cannot be run successfully; this point is also illustrated by the experiment in Section S5 in the Supplementary Materials, which considers a length 10 segment. The results show that the SISR algorithm often fails to complete even with $N = 1,000,000$ (i.e., all particles end up with weight zero), so simply increasing N in SISR does not address the weight degeneracy problem for longer segments. In contrast, UDSMC provides stable estimates with a much smaller computational budget: using $N = 5000$ and $M = 20$ works well and costs 10 times less than SISR with $N = 1,000,000$.

2.6 Example: Estimating the atomic contacts in the SARS-CoV-2 spike protein

The COVID-19 pandemic was caused by the novel coronavirus SARS-CoV-2, with the first identified outbreak in Wuhan, China, in 2019 (Chen et al., 2020). This virus binds with a human host cell via an interaction between its spike protein's receptor-binding domain (RBD) and the host cell's angiotensin-converting enzyme 2 (ACE2) receptor (Lan et al., 2020). The protein segments of the RBD involved in binding have thus been of scientific interest towards the development of therapeutics. Four key segments of amino acids have been previously identified as being involved in RBD–ACE2 binding; of these, several research groups have suggested that the segment 472–490 (known as *Loop 3*) exhibits the greatest dynamic movement (Dehury et al., 2020; Nguyen et al., 2020; Ali and Vijayan, 2020; Williams et al., 2021). The presence of such dynamic movement has implications for drug design; e.g., a drug could target a specific conformation of the segment to prevent the virus from being able to bind.

Since the initial COVID-19 outbreak, laboratory work (e.g., Wrapp et al., 2020) has provided static snapshots for the overall 3-D structure of the SARS-CoV-2 spike protein, which are publicly available in the PDB. These static snapshots have been useful as a

Figure 2.2: Estimated Boltzmann averages for the atomic contacts of C_α atoms at amino acid positions 473 to 489 of the SARS-CoV-2 spike protein, using the samples obtained from running UDSMC with $N = 500000$ and $M = 20$.



starting point for studying the dynamic movement of key segments; e.g., Williams et al. (2021) used MD simulations to explore the possible conformations of Loop 3. The importance of Loop 3 was further reinforced by the emergence of the Delta and Omicron variants of SARS-CoV-2; both of these variants had mutations (i.e., changes to the amino acid sequence) in Loop 3 that could induce changes to its conformation.

In this example, we took the same starting 3-D structure of the spike protein as Williams et al. (2021), and applied UDSMC to sample backbone conformations of Loop 3. We then estimated the Boltzmann averages for the atomic contacts for each C_α in the segment. Amino acid positions that have fewer contacts on average tend to be closer to the protein surface, and hence more likely to be directly involved in binding. We ran UDSMC with $M = 20$ and $N = 500000$, and the estimated Boltzmann averages for $n(C_\alpha^{473}), \ldots, n(C_\alpha^{489})$ are plotted in Figure 2.2. These results suggest that on average, position 478 is the most likely to be on the protein's surface (among several consecutive positions with a low number of contacts, namely 476–479), while position 483 is the least likely. Interestingly, Williams et al. (2021) also found that 476, 477, and 478, were among the top four amino acid positions where mutations occurred; from a biological point of view, this suggests that these surface amino acids are quite resilient to mutation.

2.7 Conclusion and Discussion

In this paper, we proposed an SMC method that features an upsampling-downsampling framework (UDSMC), with an emphasis on sampling backbone conformations of protein segments from the Boltzmann distribution. Existing SMC methods are not effective for this problem: SISR is hindered by severe weight degeneracy while incorporating a lookahead strategy only partially alleviates the difficulty of sampling conformations. UDSMC preserves the collection of distinct particles needed for sampling from a highly constrained distribution and thereby provides an efficient approach for estimating Boltzmann averages. We show that UDSMC generates a properly weighted sample and illustrate the performance of the method in a simulation study. As a real data example, we used UDSMC to estimate the atomic contacts for a key segment of the SARS-CoV-2 spike protein. Our work is also distinct from methods designed for protein folding; the latter usually search for conformations with the lowest energy, without regard for properly weighted samples.

UDSMC provides an approach to estimate Boltzmann averages but comes with a price: its computational complexity is $O(NM)$. Thus, choosing a value of M involves a tradeoff between computing speed and exploration effort of high-density regions. It is sensible to fix the computational budget NM before selecting M for a given application. While there is no universally optimal choice of M for all SMC applications, we can provide some intuition. When the target distribution is more uniform, a larger N is preferred; in the extreme case where the target distribution is uniform over the support, the optimal choice is clearly $M = 1$ to maximize the number of samples N for a fixed NM. In contrast, when the target distribution has many sharp local modes or is highly constrained, a larger M is preferred to ensure that UDSMC sufficiently explores the regions with positive density during propagation steps, and a specific value of M could be chosen according to Monte Carlo variance. Therefore, while UDSMC is generally applicable, its efficacy will depend on the target distribution of interest. When sampling protein segments, geometric constraints lead to a rough energy surface with many local modes and large regions of zero density. This situation is well-suited for UDSMC, whereas other SMC methods could fail.

Potential future applied work could involve protein analyses with different structural quantities or energy functions. It is straightforward to adapt UDSMC for this purpose. The sampling framework could also be extended to consider both the backbone and the side chains. One approach would be to sample the side chains after sampling backbone conformations, effectively applying SMC twice and accumulating the energy contributions.

Chapter 3

Particle Gibbs for Likelihood-Free Inference of Stochastic Volatility Models

3.1 Introduction to Financial Time Series and Volatility Modeling

State space models (SSMs) describe dynamic systems via a time series of unobserved variables and observations generated conditional on those variables (Kitagawa, 1998). The unobserved variables are commonly known as *hidden states*, and are often taken to be a discrete-time Markov process; then the SSM is specified via the transition probability of the hidden states and the likelihood of the observations. When the model also involves unknown parameters, Markov chain Monte Carlo (MCMC, Jacquier et al., 1994) and sequential Monte Carlo (SMC, Liu and West, 2001; Storvik, 2002; Carvalho et al., 2010) are general computational methods for inferring the parameters and hidden states together, by drawing samples from their joint posterior distribution in a Bayesian setting. However, the inference problem is more difficult when the likelihoods are intractable: examples are stochastic kinetic models in systems biology (Owen et al., 2015; Lowe et al., 2023) and stochastic volatility models with an intractable noise term (Vankov et al., 2019). In this likelihood-free setting, standard inference methods are no longer applicable, and we develop a new sampler to tackle this problem.

The proposed method is, in our context, illustrated in modelling stochastic volatility

in financial time series, and we start with an introduction to the background. Given a time series of observed prices $\{P_t\}_{t=0}^T$, for $t = 1, \ldots, T$ the *return* is defined as $r_t \equiv \log(P_t) - \log(P_{t-1})$ and the *volatility* is defined as $h_t \equiv Var(r_t|r_{1:t-1})$. The characteristic features of such return series are known as the stylized facts and Xie (2017) summarized the most acknowledged ones of them as follows:

- Given the past returns, the conditional expectation of the return is nearly zero;
- Given the past returns, the conditional standard deviation of the return varies over time with large values clustered;
- The return series has a heavier tail compared to the normal distribution;
- Weak correlations between pairs of returns but strong correlations between pairs of the squared returns are observed in financial time series.

Those stylized facts imply the important role of volatility in the return series, so models for the return series are often known as volatility models.

In the analysis of financial time series, volatility is commonly used to quantify uncertainty or risk. Given the wide application of volatility in finance, volatility modeling has drawn the interest of scientists since the last century. Engle (1982) proposed the autoregressive conditional heteroskedasticity (ARCH) model to describe the evolution of the conditional volatility along with the incoming observations. Bollerslev (1986) generalized the ARCH model, thus named the GARCH model. Assume a sequence of (zero-mean) returns $r_1, r_2, \ldots, r_t, \ldots$ and define the volatility h_t as $Var(r_t|r_{1:t-1})$ and $r_t = \sqrt{h_t} \cdot Z_t$ where Z_t follows some i.i.d. distribution (e.g. standard normal); then GARCH(p, q) suggests

$$h_t = \gamma + \sum_{i=1}^{q} \alpha_i r_{t-i}^2 + \sum_{j=1}^{p} \beta_j h_{t-j}.$$

Following the GARCH model, many variants of GARCH have been introduced. For example, Geweke (1986), Pantula (1986) and Milhøj (1987) proposed the log-GARCH(p, q) model as

$$\log(h_t) = \gamma + \sum_{i=1}^{q} \alpha_i \log(r_{t-i}^2) + \sum_{j=1}^{p} \beta_j \log(h_{t-j}),$$

which has some intriguing features (Francq et al. (2013) and Sucarrat (2019)) such as (i) fitted volatility is guaranteed to be positive due to the exponential specification; (ii)

no non-negativity constraints on parameters; (iii) standard inference valid under nullity of parameters; (iv) no non-negativity constraints on covariates; (v) invariance to power-transformations; (vi) robustness to outliers; (vii) generality of specification; (viii) log-GARCH models admit ARMA representations. Parameter estimation and model selection (e.g., choosing the order p, q) for those GARCH-type models can be easily computed with available software.

The GARCH-type models have deterministic evolution equations, so h_t is determined once all the past information is known. It can be extended by assuming that h_t is a random variable conditional on the past information, and it is known as *stochastic volatility*. A classic SVM (Jacquier et al., 1994, 2004; Taylor, 2008), which has been widely used in option pricing and portfolio management, adopts the form of an SSM: for $t = 1, \ldots, T$,

$$\log(h_t) = \tau + \phi \log(h_{t-1}) + \sigma_h \epsilon_t, \quad r_t = \sqrt{h_t} \times Z_t \tag{3.1}$$

where $\theta \equiv (\tau, \phi, \sigma_h^2)$ denotes the parameters, Z_t and ϵ_t are independent noise terms, and an initial distribution is assigned for h_0. In the log-volatility process, τ contributes to the mean, ϕ is an auto-regressive parameter that governs the persistence (with $|\phi| < 1$ required for second-order stationarity) (Jacquier et al., 2004), and σ_h^2 is a noise level that governs the amount of autocorrelation. Model (3.1) has a hierarchical structure: we let $l_t(r_t \mid h_t)$ denote the likelihood density, and $g_t(h_t \mid h_{t-1}, \theta)$ denote the transition density.

We now briefly discuss the inference methods for Model (3.1). In practice, Z_t and ϵ_t are often assumed to follow standard Gaussian distributions; then we can integrate out $h_{0:T}$ analytically to directly obtain the likelihood $p(r_{1:T} \mid \theta)$, and the SVM parameters can be estimated by standard methods, e.g., maximum likelihood in a frequentist approach (Fridman and Harris, 1998), and MCMC in a Bayesian approach (Jacquier et al., 1994). However, empirical studies suggest that the Gaussian assumption for Z_t may not adequately capture the heavy tails and skewness of financial time series (Engle and Patton, 2001) and Figure 3.1 presents a practical example. As a more flexible alternative, the so-called *stable distribution* (Mandelbrot, 1963; Nolan, 1997) can be adopted for Z_t instead (Lombardi and Calzolari, 2009; Vankov et al., 2019); despite its flexibility of forms for which the heavy-tailedness can be adjusted, it also has an important property called generalized central limit theorem, which states that if the summation of i.i.d. variables converges to some random variable, it will be a stable variable. However, the stable distribution does not have a closed-form density function, hence employing it for Z_t leads to an analytically intractable likelihood $l_t(r_t \mid h_t)$ that is expensive to approximate; this is the setting of interest in this paper.

Particle Markov chain Monte Carlo (PMCMC, Andrieu et al., 2010) is a general sampling approach that can be adapted to handle Model (3.1) in a Bayesian setting with

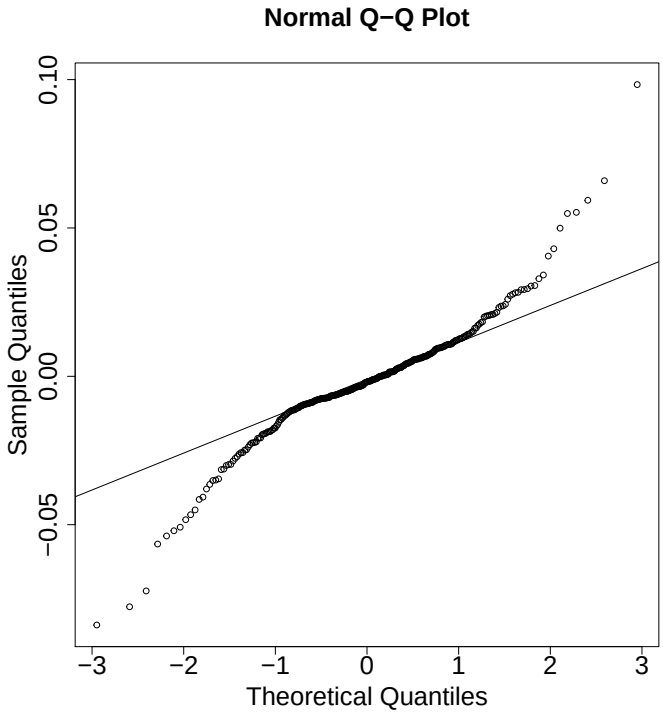


Figure 3.1: The normal Q-Q plot of the S&P 500 Index returns from January 2008 to March 2009. Both tails of the returns are much heavier than the Gaussian tails, which implies the Gaussian assumption for Z_t in Equation (3.1) is not proper and needs to be replaced with a more heavy-tailed distribution.

intractable likelihoods. Initially proposed for inference of SSMs, PMCMC provides a class of algorithms that combine the features of MCMC and SMC; in this context, SMC methods (also known as particle filters) are well-suited for drawing posterior samples of the hidden states in an SSM (e.g., $h_{0:T}$ in Model (3.1)). To our best knowledge, PMCMC is the main type of approach for SVM parameter inference when it is not possible to integrate out $h_{0:T}$ (Vankov et al., 2019). Letting $p(\theta)$ denote the joint prior for θ, PMCMC then targets the joint posterior of θ and $h_{0:T}$, namely

$$p(\theta, h_{0:T} \mid r_{1:T}) \propto p(\theta) g_0(h_0 \mid \theta) \prod_{t=1}^{T} g_t(h_t \mid h_{t-1}, \theta) l_t(r_t \mid h_t),$$

if $l_t(r_t \mid h_t)$ can be evaluated. (The corresponding version of PMCMC that bypasses this likelihood evaluation is discussed in Section 3.2.) When closed-form conditional distributions of the parameters are available, a special case of PMCMC, known as *particle Gibbs*, can be implemented. As applied here, the basic strategy of particle Gibbs alternates between sampling from $p(\theta \mid h_{0:T}, r_{1:T})$ and $p(h_{0:T} \mid \theta, r_{1:T})$ at each iteration. With the help of conjugate priors, sampling from $p(\theta \mid h_{0:T}, r_{1:T})$ can be straightforward. Sampling from $p(h_{0:T} \mid \theta, r_{1:T})$ can potentially be handled by SMC as for an SSM; e.g., one might apply SMC and obtain a particle approximation of $p(h_{0:T} \mid \theta, r_{1:T})$, denoted by $\widehat{p}_{SMC}(h_{0:T} \mid \theta, r_{1:T})$. However, it is not valid to simply substitute sampling from $p(h_{0:T} \mid \theta, r_{1:T})$ with sampling from $\widehat{p}_{SMC}(h_{0:T} \mid \theta, r_{1:T})$ in particle Gibbs, because doing so does not admit the target distribution as invariant (Andrieu et al., 2010). To correctly embed SMC within a particle Gibbs algorithm, the form of SMC known as *conditional SMC* (cSMC) should be implemented instead. The fundamental idea of cSMC is to use a trajectory as input for regular SMC, and this input trajectory is preserved in all resampling steps; we review the details of cSMC algorithms in Section 3.2.2.

Approximate Bayesian computation (ABC) is a general technique that can be used to bypass the evaluation of an intractable or expensive likelihood, if one can directly simulate observations from the likelihood (Marin et al., 2012). Thus, to perform inference on Model (3.1) with an analytically intractable $l_t(r_t \mid h_t)$, ABC may be combined with cSMC. The basic idea is to introduce a sequence of auxiliary observations (that are sampled from the likelihood) and then assign weights according to the "distances" between the auxiliary observations and the actual observations; ABC-based methods are further reviewed in Section 3.2.3. It is straightforward to construct ABC versions of existing cSMC algorithms based on the bootstrap filter (BF, Gordon et al., 1993); however, in practice the particles they generate can tend to have many near-zero weights due to large "distances", i.e., the algorithms suffer from weight degeneracy. In the SMC literature, specific techniques to mitigate weight degeneracy include the SMC sampler with annealed importance sampling (Del Moral et al., 2006), weight tempering via lookahead strategies (Lin et al., 2013), and drawing multiple descendants per particle (Hou and Wong, 2024); however, as cSMC must be embedded within every iteration of particle Gibbs, these techniques would be computationally expensive to apply. In contrast, the auxiliary particle filter (APF, Pitt and Shephard, 1999) is a common alternative to the BF that can help reduce weight degeneracy, since the resampling step of the APF accounts for the one-step-ahead observation.

In this paper, we use the APF as a strategy to reduce weight degeneracy and improve parameter estimation in the particle Gibbs and ABC setting. In related work, Vankov et al. (2019) proposed an ABC-based APF that uses auxiliary observations to assign importance weights for PMCMC. However, that ABC-based APF has only been

embedded within a particle Metropolis-Hastings algorithm; a Gibbs sampler is preferred over Metropolis-Hastings proposals when closed-form conditional distributions are available for the parameters. Therefore, as the main contribution of this paper, we propose to embed an ABC-based APF within a particle Gibbs algorithm. We show that our proposed algorithm satisfies the form of cSMC, and thus admits the target posterior distribution as invariant. We then perform inference on Model (3.1) when Z_t is assumed to follow the stable distribution. The results demonstrate that our particle Gibbs algorithm significantly outperforms existing ones, thereby providing a useful computational method for handling SSMs with intractable likelihoods.

The paper is organized as follows: in Section 3.2, we review ABC and particle Gibbs methods, and present the proposed ABC-based particle Gibbs algorithm with a conditional auxiliary particle filter (ABC-PG-cAPF). In Section 3.3, we illustrate the effectiveness of the proposed sampler for inference of Model (3.1) with the stable distribution. In Section 3.4, we present a real data application by fitting an SVM to S&P 500 daily returns during the 2008–2009 financial crisis. In Section 3.5, we briefly summarize the paper and its contributions and discuss some potential future directions.

3.2 Methodology

3.2.1 Model setup

The *stable distribution* is defined via its characteristic function (Nolan, 2020): we denote $Z \sim SD(\alpha, \beta, \gamma, \delta)$ if Z follows a stable distribution with parameters $\alpha \in (0, 2]$ for heavy-tailedness, $\beta \in [-1, 1]$ for skewness, $\gamma \in [0, \infty)$ for scale and $\delta \in (-\infty, \infty)$ for location, and Z has characteristic function

$$\psi_Z(t) = \begin{cases} \exp\left(-\gamma^\alpha |t|^\alpha \left[1 + i\beta \operatorname{sign}(t) \tan\left(\frac{\pi\alpha}{2}\right)\left\{(\gamma|t|)^{1-\alpha} - 1\right\}\right] + i\delta t\right) & \text{if } \alpha \neq 1 \\ \exp\left(-\gamma|t|\left\{1 + \frac{2}{\pi} i\beta \operatorname{sign}(t) \log(\gamma|t|)\right\} + i\delta t\right) & \text{if } \alpha = 1. \end{cases}$$

Its corresponding density function, $f_Z(z) = \frac{1}{2\pi}\int_{-\infty}^{\infty} \psi_Z(t)\exp(-izt)dt$, does not have an analytical form in general; some basic facts are that if $\alpha > 1$, $E[Z] = \delta - \beta\gamma\tan(\pi\alpha/2)$ (and undefined otherwise); if $\alpha < 2$, $Var(Z) = \infty$ (Mandelbrot, 1963; Nolan, 1997, 2020). Evaluation of $f_Z(z)$ is expensive and commonly needs to be approximated by the fast

Fourier transform (Mittnik et al., 1999) or numerical integration (Nolan, 1997, 1999), but it is feasible to simulate realizations of Z according to $\psi_Z(t)$ based on the work of Kanter (1975) and Chambers et al. (1976); in this paper, we use the R package *stabledist* (Wuertz et al., 2016) for generating random draws of Z.

The main model of interest in this paper is then the SVM specified in (3.1) with $\epsilon_t \sim N(0,1)$ and $Z_t \sim SD(\alpha, \beta, \gamma, \delta)$ for $t = 1, \ldots, T$, all independent. We also set $h_0 \sim \text{Log-normal}(\tau/(1-\phi), \sigma_h^2/(1-\phi^2))$ according to the stationarity of the log-volatility process. The goal is to infer the parameters $\theta = (\tau, \phi, \sigma_h^2)$ given a time series of returns r_t. We take a Bayesian approach to inference and assign conjugate priors for the parameters:

$$\sigma_h^2 \sim IG(a_0, b_0), \quad (\tau, \phi) \mid \sigma_h^2 \sim N(\boldsymbol{\mu}_0, \sigma_h^2 \boldsymbol{\Lambda}_0^{-1}) \text{ with } |\phi| < 1$$

i.e., the joint prior for θ is a truncated normal–inverse Gamma with hyperparameters $a_0, b_0, \boldsymbol{\mu}_0, \boldsymbol{\Lambda}_0$, which we denote as $NIG(a_0, b_0, \boldsymbol{\mu}_0, \boldsymbol{\Lambda}_0)$; the truncated support for ϕ ensures the required second-order stationarity of the log-volatility process. The conjugacy of these priors follows from the results of Bayesian linear regression, i.e., by defining

$$\mathbf{X} = \begin{bmatrix} 1 & \log(h_0) \\ \vdots & \vdots \\ 1 & \log(h_{T-1}) \end{bmatrix} \text{ and } \boldsymbol{y} = \begin{bmatrix} \log(h_1) \\ \vdots \\ \log(h_T) \end{bmatrix},$$

then $p(\theta \mid h_{0:T}, r_{1:T}) = p(\theta \mid h_{0:T})$ is a truncated $NIG(a_T, b_T, \boldsymbol{\mu}_T, \boldsymbol{\Lambda}_T)$ with $|\phi| < 1$, where

$$\begin{aligned} \boldsymbol{\Lambda}_T &= \left(\mathbf{X}^{\mathrm{T}}\mathbf{X} + \boldsymbol{\Lambda}_0\right), \quad \boldsymbol{\mu}_T = \boldsymbol{\Lambda}_T^{-1}\left(\boldsymbol{\Lambda}_0\boldsymbol{\mu}_0 + \mathbf{X}^{\mathrm{T}}\boldsymbol{y}\right), \\ a_T &= a_0 + \frac{T}{2}, \quad b_T = b_0 + \frac{1}{2}\left(\mathbf{y}^{\mathrm{T}}\mathbf{y} + \boldsymbol{\mu}_0^{\mathrm{T}}\boldsymbol{\Lambda}_0\boldsymbol{\mu}_0 - \boldsymbol{\mu}_T^{\mathrm{T}}\boldsymbol{\Lambda}_T\boldsymbol{\mu}_T\right). \end{aligned}$$

Thus, a Gibbs sampler update of θ only depends on these sufficient statistics. The choice of hyperparameters could be informed by previous empirical studies, or set to resemble a flat or weakly informative prior.

3.2.2 Particle Gibbs for the SVM with tractable likelihoods

In the SVM context, a particle Gibbs algorithm alternates between sampling from $p(\theta \mid h_{0:T}, r_{1:T})$ and $p(h_{0:T} \mid \theta, r_{1:T})$. As described in the Introduction, sampling from $p(h_{0:T} \mid \theta, r_{1:T})$ requires the use of conditional SMC algorithms. The basic idea of cSMC is to take an input trajectory (i.e., a sample for $h_{0:T}$) as a reference and produce an output trajectory via SMC-style propagation; furthermore, cSMC accounts for all random variables generated

Algorithm 2: Conditional Bootstrap Filter

input: observations $r_{1:T}$, input trajectory $h^*_{0:T}$, transition density g_t, likelihood l_t;
draw $h_0^{(n)} \sim g_0(h_0)$ for $n = 1, \ldots, N-1$ and set $h_0^{(N)} = h_0^*$;
$w_0^{(n)} = \frac{1}{N}$ for $n = 1, \ldots, N$;
for *t in 1:T* **do**
  draw index $a_{t-1}^{(n)}$ from $\{(n, w_{t-1}^{(n)})\}_{n=1}^N$ for $n = 1, \ldots, N-1$;
  set index $a_{t-1}^{(N)} = N$;
  draw $h_t^{(n)} \sim g_t(h_t \mid h_{t-1}^{(a_{t-1}^{(n)})})$ for $n = 1, \ldots, N-1$ and set $h_t^{(N)} = h_t^*$;
  $w_t^{(n)} = l_t(r_t \mid h_t^{(n)})$ for $n = 1, \ldots, N$;
end
draw index b from $\{(n, w_T^{(n)})\}_{n=1}^N$;
return $h_{0:T}^{(b)}$;

during propagation via an extended target distribution of higher dimension (Chopin and Singh, 2015). The steps for a sweep of particle Gibbs then consist of (i) updating the parameters based on the input trajectory, (ii) generating new trajectories based on the updated parameters and input trajectory, and (iii) selecting an output trajectory as input for the next iteration (Andrieu et al., 2010). The use of cSMC in particle Gibbs guarantees the target distribution is admitted as the invariant density. Here, we briefly review two existing cSMC algorithms that are applicable within particle Gibbs when the likelihood $l_t(r_t \mid h_t)$ can be computed.

The first is the conditional bootstrap filter (cBF, Andrieu et al., 2010) as summarized in Algorithm 2. A key feature is that it preserves the input trajectory $h^*_{0:T}$ throughout propagation and resampling; holding $h^*_{0:T}$ intact, $N-1$ new trajectories are generated "conditional on" $h^*_{0:T}$ in SMC fashion; finally one of the N trajectories (i.e., among the input trajectory and the $N-1$ generated trajectories) is selected to be the new input trajectory for the next iteration. As shown in Algorithm 2, for any particle $h_t^{(n)}$ with $n \in \{1, \ldots, N\}$ simulated at step t, we denote the index of the ancestor particle of $h_t^{(n)}$ by $a_{t-1}^{(n)}$, i.e., $h_t^{(n)}$ is propagated from $h_{t-1}^{(a_{t-1}^{(n)})}$. For simplicity, let $A_{t,t}^{(n)} = n$, $A_{t-1,t}^{(n)} = a_{t-1}^{(n)}$ and $A_{t-l,t}^{(n)} = a_{t-l}^{(A_{t-l+1,t}^{(n)})}$ for $2 \le l \le t$; then the n-th trajectory can be written as $h_{0:T}^{(n)} = (h_0^{(A_{0,T}^{(n)})}, h_1^{(A_{1,T}^{(n)})}, \ldots, h_T^{(A_{T,T}^{(n)})})$ after $t = T$ propagation and resampling steps.

The second is the conditional bootstrap filter with ancestor sampling (cBFAS, Lindsten

Algorithm 3: Conditional Bootstrap Filter with Ancestor Sampling

input: observations $r_{1:T}$, input trajectory $h^*_{0:T}$, transition density g_t, likelihood l_t;
draw $h_0^{(n)} \sim g_0(h_0)$ for $n = 1, \ldots, N-1$ and set $h_0^{(N)} = h_0^*$;
$w_0^{(n)} = \frac{1}{N}$ for $n = 1, \ldots, N$;
for *t in 1:T* **do**
  draw index $a_{t-1}^{(n)}$ from $\{(n, w_{t-1}^{(n)})\}_{n=1}^N$ for $n = 1, \ldots, N-1$;
  draw index $a_{t-1}^{(N)}$ from $\{(n, w_{t-1}^{(n)} g_t(h_t^* \mid h_{t-1}^{(n)}))\}_{n=1}^N$;
  draw $h_t^{(n)} \sim g_t(h_t \mid h_{t-1}^{(a_{t-1}^{(n)})})$ for $n = 1, \ldots, N-1$ and set $h_t^{(N)} = h_t^*$;
  $w_t^{(n)} = l_t(r_t \mid h_t^{(n)})$ for $n = 1, \ldots, N$;
end
draw index b from $\{(n, w_T^{(n)})\}_{n=1}^N$;
return $h_{0:T}^{(b)}$;

et al., 2014) as summarized in Algorithm 3. The cBF keeps the input trajectory $h^*_{0:T}$ intact, i.e., $A_{t,T}^{(N)} = N$, and preserved throughout for all $t \in \{0, \ldots, T-1\}$, so the early parts of generated trajectories may tend to closely resemble (or be identical to) the input trajectory, which when too extreme is known as path degeneracy. The key idea of cBFAS is to stochastically perturb the input trajectory by breaking it into pieces via ancestor sampling. Ancestor sampling, as presented in Algorithm 3, takes $A_{t,T}^{(N)}$ to be stochastic, so the input trajectory can be partially replaced by other generated trajectories at each step t. Consequently, the input trajectory interacts much more with the other trajectories (Svensson et al., 2015), which can help cBFAS mitigate path degeneracy while maintaining the target distribution as invariant (Lindsten et al., 2014; Svensson et al., 2015).

3.2.3 Review of ABC methods and ABC-SMC methods

To briefly review ABC methods, consider observations r and parameters θ where the likelihood $l(r \mid \theta)$ is intractable and expensive to evaluate. Consequently, the posterior $p(\theta \mid r) \propto p(\theta) l(r \mid \theta)$ is also intractable. In this setting, ABC can be used for inference of θ if sampling from $l(\cdot \mid \theta)$ is straightforward. The basic idea of ABC methods is to construct an approximation to the posterior with the help of auxiliary observations (denoted by u) and an ABC kernel (denoted by $K_\epsilon(r \mid u)$). The simplest example of an ABC method is a likelihood-free rejection sampling algorithm with a uniform kernel (i.e., $K_\epsilon(r \mid u) \propto \mathbb{1}_{u-\epsilon<r<u+\epsilon}$ for a chosen $\epsilon > 0$) that implements the following steps:

(1) sample a candidate θ^* from the prior $p(\theta)$; (2) sample a realization u from $l(u \mid \theta^*)$; (3) accept (θ^*, u) if $|r - u| < \epsilon$. Then accepted samples of (θ^*, u) follow the density defined by $p_\epsilon(\theta, u \mid r) \propto p(\theta) l(u \mid \theta) K_\epsilon(r \mid u)$, which can be viewed as the ABC approximation of the extended distribution $p(\theta, u \mid r)$. In practice, Gaussian kernels (i.e., $K_\epsilon(r \mid u) \propto \exp\left[(r-u)^2/(2\epsilon^2)\right]$) are more commonly used than uniform ones. It is clear that $p_\epsilon(\theta \mid r) \xrightarrow{\mathrm{d}} p(\theta \mid r)$ as $\epsilon \to 0$ and $p_\epsilon(\theta \mid r) \xrightarrow{\mathrm{d}} p(\theta)$ as $\epsilon \to \infty$. Therefore, an ABC kernel with a small ϵ can lead to many near-zero weights of generated candidates (θ^*, u) (or more rejected samples), while a larger ϵ can lead to more uniform weights (or more accepted samples); however, the accuracy of the ABC approximation will decrease as a tradeoff.

For an SSM with intractable likelihoods, we can similarly construct the ABC approximation $p_\epsilon(h_{0:T}, u_{1:T} \mid r_{1:T}, \theta)$ of the corresponding extended distribution and sample from it using an SMC algorithm; this is known as ABC-SMC (Peters et al., 2012). Thus, the target distribution of ABC-SMC involves the auxiliary observations $u_{1:T}$ for evaluating importance weights and has the form

$$p_\epsilon(h_{0:T}, u_{1:T} \mid r_{1:T}, \theta) \propto g_0(h_0 \mid \theta) \prod_{t=1}^{T} g_t(h_t \mid h_{t-1}, \theta) K_\epsilon(r_t \mid u_t) l_t(u_t \mid h_t). \tag{3.2}$$

To sample from (3.2), Vankov et al. (2019) proposed an ABC-based APF, which is summarized in Algorithm 4 and applicable within a particle Metropolis-Hastings algorithm. Within a particle Gibbs algorithm, however, Algorithm 4 cannot be directly used (recall that particle Gibbs requires a cSMC setup to admit the target distribution as invariant). Thus, in the following we propose a new ABC-based APF, which we call the ABC-based conditional auxiliary particle filter (ABC-cAPF), that can be embedded within a particle Gibbs algorithm.

3.2.4 Likelihood-free ABC-based cSMC for the SVM

Now we consider the SVM with stable distribution as presented in Section 3.2.1. In the ABC setting with a chosen kernel K_ϵ, Model (3.1) can then be rewritten as

$$\log(h_t) = \tau + \phi \log(h_{t-1}) + \sigma_h \epsilon_t, \quad u_t = \sqrt{h_t} \times Z_t, \quad r_t \sim K_\epsilon(\cdot \mid u_t) \tag{3.3}$$

with $\epsilon_t \sim N(0,1)$ and $Z_t \sim SD(\alpha, \beta, \gamma, \delta)$ all independent for $t = 1 \ldots, T$, and $h_0 \sim$ Log-normal$(\tau/(1-\phi), \sigma_h^2/(1-\phi^2))$. Here, each h_t, $t = 1 \ldots, T$ corresponds to an auxiliary

Algorithm 4: ABC-based Auxiliary Particle Filter

input: observations $r_{1:T}$, input trajectory $h^*_{0:T}$, transition density g_t, likelihood l_t;
draw $h_0^{(n)} \sim g_0(h_0)$ for $n = 1, \ldots, N-1$ and set $h_0^{(N)} = h_0^*$;
$w_0^{(n)} = \frac{1}{N}$ for $n = 1, \ldots, N$;
for *t in 1:T* **do**
  for *n in 1:N* **do**
    $\widetilde{w}_{t-1}^{(n)} = w_{t-1}^{(n)} \int\int K_\epsilon(r_t \mid u_t) l_t(u_t \mid h_t) g_t(h_t \mid h_{t-1}^{(n)}) dh_t du_t$;
    draw index $a_{t-1}^{(n)}$ from $\{(n, \widetilde{w}_{t-1}^{(n)})\}_{n=1}^N$;
    draw $h_t^{(n)} \sim g_t(h_t \mid h_{t-1}^{(a_{t-1}^{(n)})})$;
    draw $u_t^{(n)} \sim l_t(u_t \mid h_t^{(n)})$;
    $w_t^{(n)} = \frac{w_{t-1}^{(a_{t-1}^{(n)})}}{\widetilde{w}_{t-1}^{(a_{t-1}^{(n)})}} K_\epsilon(r_t \mid u_t^{(n)})$;
  end
end
draw index b from $\{(n, w_T^{(n)})\}_{n=1}^N$;
return $h_{0:T}^{(b)}$;

observation u_t for importance weight calculation according to K_ϵ. The extended posterior distribution of interest is then

$$p_\epsilon(\theta, h_{0:T}, u_{1:T} \mid r_{1:T}) \propto p(\theta) g_0(h_0 \mid \theta) \prod_{t=1}^{T} g_t(h_t \mid h_{t-1}, \theta) K_\epsilon(y_t \mid u_t) l_t(u_t \mid h_t), \tag{3.4}$$

and the goal is to sample from (3.4) in a likelihood-free manner, i.e., without computing l_t. We shall focus on the particle Gibbs case, and develop ABC-based algorithms to sample from (3.4) that alternate between sampling from $p(\theta \mid h_{0:T}, u_{1:T}, r_{1:T}) = p(\theta \mid h_{0:T})$ and $p_\epsilon(h_{0:T}, u_{1:T} \mid r_{1:T}, \theta)$.

The cBF and cBFAS algorithms can be applied in the ABC setting with slight modifications: in the step where we draw each $h_t^{(n)} \sim g_t(h_t \mid h_{t-1}^{(a_{t-1}^{(n)})})$, we also draw an auxiliary observation $u_t^{(n)}$ from $l_t(u_t \mid h_t^{(n)})$ and then assign the importance weight $w_t^{(n)} = K_\epsilon(r_t \mid u_t^{(n)})$ for the particle $(h_t^{(n)}, u_t^{(n)})$. We shall call these algorithms the ABC-based conditional bootstrap filter (ABC-cBF) and ABC-based conditional bootstrap filter with ancestor sampling (ABC-cBFAS), respectively. However, these two algorithms can encounter severe weight

degeneracy in practice, if we choose a small ϵ in the ABC kernel to obtain an accurate approximation of the true target distribution. Thus, in the following we also propose a novel ABC-based cSMC algorithm using the auxiliary particle filter as the building block.

We shall call this third algorithm the ABC-based conditional auxiliary particle filter (ABC-cAPF), as presented in Algorithm 5. To initialize the ABC-cAPF at $t = 0$, $N-1$ particles, $\{h_0^{(n)}\}_{n=1}^{N-1}$, are sampled from the log-normal distribution and $h_0^{(N)}$ is assigned the initial log-volatility of the input trajectory. Then after the $(t-1)$-th propagation step $(t = 1, \ldots, T)$, the N-th particle is set to be the input trajectory $h_{0:t-1}^*$ and the remaining $N-1$ particles $\{(h_{0:t-1}^{(n)}, u_{1:t-1}^{(n)})\}_{n=1}^{N-1}$ are generated with weights that satisfy

$$w_{t-1}^{(n)} \propto g_0(h_0^{(A_{0,t-1}^{(n)})}) \prod_{s=1}^{t-1} K_\epsilon(r_s \mid u_s^{(A_{s,t-1}^{(n)})}) l_s(u_s^{(A_{s,t-1}^{(n)})} \mid h_s^{(A_{s,t-1}^{(n)})}) g_s(h_s^{(A_{s,t-1}^{(n)})} \mid h_{s-1}^{(A_{s-1,t-1}^{(n)})})$$

for $n = 1, \ldots, N-1$. Following the concept of properly weighted particles in SMC (Liu, 2001), this weight ensures that the set of $N-1$ generated particles is properly weighted with respect to $p_\epsilon(h_{0:t-1}, u_{1:t-1} \mid r_{1:t-1}, \theta)$. As an important feature of the APF (Pitt and Shephard, 1999) when propagating the particles from $t-1$ to t, the tempered weights $\widetilde{w}_{t-1}^{(n)}$, i.e., the adjusted importance weights that incorporate the one-step-ahead observation r_t, are computed and utilized; in the ABC setting, the computation of these tempered weights require an intractable integration, namely

$$\widetilde{w}_{t-1}^{(n)} = w_{t-1}^{(n)} p(r_t \mid h_{t-1}) = w_{t-1}^{(n)} \int \int K_\epsilon(r_t \mid u_t) l_t(u_t \mid h_t) g_t(h_t \mid h_{t-1}^{(n)}) dh_t du_t \tag{3.5}$$

where the double integral does not have a closed form; Vankov et al. (2019) suggest approximating it with a more heavy-tailed distribution such as a t-class distribution.

Specifically, given $r_t = \sqrt{h_t} \times Z_{\text{stable}}$, we construct an approximation of the stable distribution noise with a standard Cauchy noise as $\log(r_t^2) = \log(h_t) + \log(Z_{\text{Cauchy}}^2)$ (thus $E\{\log(r_t^2) \mid h_{t-1}\} = \tau + \phi \log(h_{t-1})$ and $Var\{\log(r_t^2) \mid h_{t-1}\} = \sigma_h^2 + \pi^2$). Then we approximate $\log(r_t^2)$ with a linear combination of $\log(Z_{\text{Cauchy}}^2)$ and $\log(h_{t-1})$ to match the conditional mean and variance, which gives $\log(Z_{\text{cauchy}}^2) = \sqrt{\frac{\pi^2}{\sigma_h^2+\pi^2}}\{\log(r_t^2) - \tau - \phi \log(h_{t-1})\}$, so $p(r_t \mid h_{t-1})$ is approximately proportional to

$$\left(1 + (r_t^2)^{\sqrt{\frac{\pi^2}{\sigma_h^2+\pi^2}}} \exp\left[-\sqrt{\frac{\pi^2}{\sigma_h^2+\pi^2}}\{\tau + \phi \log(h_{t-1})\}\right]\right)^{-1}.$$

Note that the quality of the integral approximation in (3.5) does not influence the validity of ABC-cAPF; however, the tempered weights should cover the high-density regions of the true importance weights for a more efficient algorithm.

Algorithm 5: ABC-based Conditional Auxiliary Particle Filter

input: observations $r_{1:T}$, input trajectory $h^*_{0:T}$, transition density g_t, likelihood l_t;
draw $h_0^{(n)} \sim g_0(h_0)$ for $n = 1, \ldots, N-1$ and set $h_0^{(N)} = h_0^*$;
$w_0^{(n)} = \frac{1}{N}$ for $n = 1, \ldots, N$;
for *t in 1:T* **do**
  compute $\widetilde{w}_{t-1}^{(n)}$ using $w_{t-1}^{(n)}$, r_t and $h_{t-1}^{(n)}$ for $n = 1, \ldots, N$;
  draw index $a_{t-1}^{(n)}$ from $\{(n, \widetilde{w}_{t-1}^{(n)})\}_{n=1}^N$ for $n = 1, \ldots, N-1$ and set index $a_{t-1}^{(N)} = N$;
  draw $h_t^{(n)} \sim g_t(h_t \mid h_{t-1}^{(a_{t-1}^{(n)})})$ for $n = 1, \ldots, N-1$ and set $h_t^{(N)} = h_t^*$;
  draw $u_t^{(n)} \sim l_t(u_t \mid h_t^{(n)})$ for $n = 1, \ldots, N$;
  $w_t^{(n)} = \frac{w_{t-1}^{(a_{t-1}^{(n)})}}{\widetilde{w}_{t-1}^{(a_{t-1}^{(n)})}} K_\epsilon(r_t \mid u_t^{(n)})$ for $n = 1, \ldots, N-1$ and $w_t^{(N)} = K_\epsilon(r_t \mid u_t^{(n)})$;
end
draw index b from $\{(n, w_T^{(n)})\}_{n=1}^N$;
return $h_{0:T}^{(b)}$;

For $t = 1, \ldots, T$, resampling and propagation are implemented with the tempered weights, but otherwise similar to the cBF: (i) resample the first $N-1$ particles from $\{(h_{0:t-1}^{(n)}, u_{1:t-1}^{(n)})\}_{n=1}^N$ proportional to the tempered weights $\{\widetilde{w}_{t-1}^{(n)}\}_{n=1}^N$ while leaving the N-th particle intact as the input trajectory; (ii) propagate the resampled particles as in the cBF; (iii) compute the importance weights of the propagated particles so that cAPF targets the same density as cBF, i.e., the weights of the propagated particles $\{(h_{0:t}^{(n)}, u_{1:t}^{(n)})\}_{n=1}^{N-1}$ satisfy

$$w_t^{(n)} \propto g_0(h_0^{(A_{0,t}^{(n)})}) \prod_{s=1}^{t} K_\epsilon(r_s \mid u_s^{(A_{s,t}^{(n)})}) l_s(u_s^{(A_{s,t}^{(n)})} \mid h_s^{(A_{s,t}^{(n)})}) g_s(h_s^{(A_{s,t}^{(n)})} \mid h_{s-1}^{(A_{s-1,t}^{(n)})}).$$

Each of the three algorithms, i.e., ABC-cBF, ABC-cBFAS, and ABC-cAPF, can be combined with a Gibbs sampler for the parameters $p(\theta \mid h_{0:T}, u_{1:T}, r_{1:T})$ to construct particle Gibbs samplers for (3.4). We call these algorithms the ABC-based particle Gibbs with ABC-cBF (ABC-PG-cBF), ABC-based particle Gibbs with ABC-cBFAS (ABC-PG-cBFAS), and ABC-based particle Gibbs with the conditional auxiliary particle filter (ABC-PG-cAPF), respectively. Algorithm 6 shows how the ABC-cAPF is embedded within the particle Gibbs sampler: each Gibbs sweep uses the ABC-cAPF to draw an output trajectory for $h_{0:T}$ (given the current draw of θ and the input trajectory), and then draws

Algorithm 6: ABC-based Particle Gibbs with the Conditional Auxiliary Particle Filter

initialization: $\theta[0]$, input trajectory $h^*_{0:T}[0]$, burning size S, sample size N
for *t in 1:*$(S+N)$ **do**
  run Algorithm 5 with $\theta[t-1]$ and $h^*_{0:T}[t-1]$ and output $h^*_{0:T}[t]$;
  update the sufficient statistics with $h^*_{0:T}[t]$;
  sample $\theta[t]$ from the posterior $NIG(a_T, b_T, \boldsymbol{\mu}_T, \boldsymbol{\Lambda}_T)$;
end
return $\theta[(S+1):N]$;

an updated θ from its closed-form NIG conditional posterior (given the output trajectory, which becomes the input trajectory for the next iteration). The ABC-PG-cBF and ABC-PG-cBFAS algorithms are constructed analogously. To initialize the particle Gibbs sampler, we draw parameters from $p(\theta)$ and an input trajectory $h^*_{0:T}$ from the log-normal distribution given those parameters.

Proposition 3. *The proposed particle Gibbs algorithm (ABC-PG-cAPF in Algorithm 6) admits the target distribution, i.e., the posterior distribution* $p_\epsilon(h_{0:T}, u_{1:T}, \theta \mid r_{1:T})$ *in* (3.4)*, as the invariant density under some mild assumptions.*

Proof. See Section S1 of the Supplementary Material. □

While ABC-PG-cAPF is proposed with Model (3.3) as the focus in this paper, the computational strategy is not limited to this specific problem and can be applied to other SSMs with intractable likelihoods, e.g., stochastic kinetic models (Owen et al., 2015; Lowe et al., 2023) and models with likelihoods that follow g-and-k distributions (Rayner and MacGillivray, 2002; Drovandi and Pettitt, 2011).

3.3 Simulation study

We consider the model described in Equation (3.3) with a second-order stationary log-volatility process. Since h_t conditional on h_{t-1} follows a log-normal distribution, we have $E(h_t) = \exp\left\{\frac{\gamma}{1-\phi} + \frac{\sigma_h^2}{2(1-\phi^2)}\right\}$ and $Var(h_t) = E^2(h_t)\left\{\exp\left(\frac{\sigma_h^2}{1-\phi^2}\right) - 1\right\}$. Therefore, the (squared) coefficient of variation, $\text{CV} = Var(h_t)/E^2(h_t)$, can be written as $\text{CV} = \exp\left(\frac{\sigma_h^2}{1-\phi^2}\right) - 1$. Practical ranges of ϕ that have been suggested from empirical analyses

are $\phi \in [0.9, 0.98]$, or more loosely, $\phi \in [0.8, 0.995]$ (Jacquier et al., 1994); following their simulation study we take $\phi \in \{0.9, 0.95, 0.98\}$, CV $\in \{0.1, 1, 10\}$, and set $E(h_t) = 0.0009$. Given the values of ϕ, CV, and $E(h_t)$, the corresponding values of γ and σ_h^2 can be easily computed. Following the setup for $SD(\alpha, \beta, \gamma, \delta)$ in Vankov et al. (2019), we hold $\gamma = 1$ and $\delta = 0$ fixed throughout and consider $(\alpha, \beta) = (1.75, 0.1), (1.7, 0.3), (1.5, -0.3)$ respectively as three experimental settings; (1.75, 0.1) gives the least heavy-tailed stable distribution and the least (right) skewness and is expected to be the easiest to handle; (1.5, -0.3) gives the most heavy-tailed stable distribution and the most (left) skewness and is expected to be the hardest to handle; (1.7, 0.3) has heavy-tailedness between that of the above two cases and the most (right) skewness. Following Yang et al. (2018), we assign a weakly informative NIG conjugate prior for θ with $a_0 = 2$, $b_0 = 0.5$, $\boldsymbol{\mu}_0 = \begin{bmatrix} 0 \\ 0.9 \end{bmatrix}$ and $\boldsymbol{\Lambda}_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$.

For each combination of ϕ, CV, and (α, β), we simulated a time series of length $T = 100$ observations. We then ran each of the particle Gibbs samplers (ABC-PG-cBF, ABC-PG-cBFAS, and ABC-PG-cAPF) using $N = 100$ particles for the embedded cSMC algorithm and a Gaussian kernel with $\epsilon = 0.001$. We discarded the first 2000 iterations as burn-in and took the 5000 iterations after the burn-in period as the posterior sample. The posterior means are treated as the Bayes' estimates for the parameters. To compare the performances of the different PG samplers, we repeated this procedure for 100 simulated datasets. The simulation results, summarized by computing the root-mean-squared errors (RMSEs) of the parameter estimates over the 100 repetitions, are presented in Tables 3.1, 3.2, and 3.3 for $(\alpha, \beta) = (1.75, 0.1), (1.7, 0.3), (1.5, -0.3)$, respectively.

The numerical results show that the proposed ABC-PG-cAPF outperforms the corresponding cBF-based algorithms with parameter RMSEs that are ∼30–70% smaller, depending on the scenario considered. The Gaussian ABC kernel tends to produce very uneven particle weights and the cBF-based algorithms become hindered by weight degeneracy. While ancestor sampling (in cBFAS) is designed to tackle path degeneracy, doing so has limited effectiveness for weight degeneracy and appears to be slightly worse than cBF in these scenarios. In contrast, the weight tempering strategy of the cAPF helps reduce the variability in the weights, which leads to more plausible trajectories being sampled, and in turn improves the accuracy of the parameter estimates.

Some specific patterns in the results can be noted: the RMSEs tend to be higher for a smaller σ_h^2 (i.e., a larger ϕ or a smaller CV) or a more heavy-tailed Z_t (i.e., a smaller α). Intuitively, volatility is more persistent for a smaller σ_h^2; however, r_t can still have large fluctuations due to the heavy tails of Z_t, so it may be difficult in practice to disentangle the variability coming from Z_t and h_t. More heavy-tailed Z_t's make the estimation problem even more challenging. Finally in the ABC setting, a smaller ϵ is generally preferred for a

better ABC approximation of the target distribution; however, the cSMC algorithms may suffer from more severe weight degeneracy when ϵ is very small. When we decreased ϵ to 0.0005, the parameter RMSEs were worse for all three algorithms, but ABC-PG-cAPF was the most robust; see Section S2 in the Supplementary Material for details.

3.4 Application: the S&P 500 Index during the Financial Crisis in 2008

We now apply the ABC-PG-cAPF algorithm to fit an SVM to time-series data obtained from the Standard & Poor 500 (S&P 500) index. Given the time-series of the daily price (i.e., the average of the open and close price on each day), we computed the daily returns for the period January 2008 to March 2009. These are displayed via the black line in the left panel of Figure 3.2. The large fluctuations around October 2008 indicate the climax of the well-known global financial crisis. Following the suggestion of Kabašinskas et al. (2009) to take stable distribution parameters in the ranges $\alpha \in (1.65, 1.8)$ and $\beta \in (-0.017, 0.2)$ for financial data, we set α and β to be the midpoints of these ranges for our analysis.

We estimated the SVM parameters based on these data, running 2000 burn-in iterations of ABC-PG-cAPF followed by 5000 sampling iterations, using $N = 500$ particles and a Gaussian ABC kernel with $\epsilon = 0.001$. The Bayes' estimates (posterior means) and credible intervals of the parameters are reported in Table 3.4. In the right panel of Figure 3.2, we plot the fitted daily volatility $\{h_t^*\}_{t=1}^T$ along with a central 95% credible interval based on the posterior samples; the estimated volatility peaks at the climax of the financial crisis. Lastly, the estimated 95% credible intervals for the returns are also superimposed on the left panel, which were computed via the samples generated from $l(r_t \mid h_t^*)$ for each draw of h_t^* at each time $t = 1, \ldots, T$ and indicate a good fit to the data.

3.5 Conclusion and Discussion

In this paper, we proposed an ABC-based cAPF embedded within a particle Gibbs sampler for likelihood-free inference of the SVM. Our proposed sampler builds upon a rich SMC and SVM literature, e.g., the idea of using MCMC for parameter estimation in SVMs (Jacquier et al., 1994), particle Gibbs samplers for state space models (Andrieu et al., 2010), and ABC-based PMCMC for SVM inference with intractable likelihoods (Vankov et al., 2019). Compared to existing particle Gibbs samplers, the proposed ABC-PG-cAPF algorithm

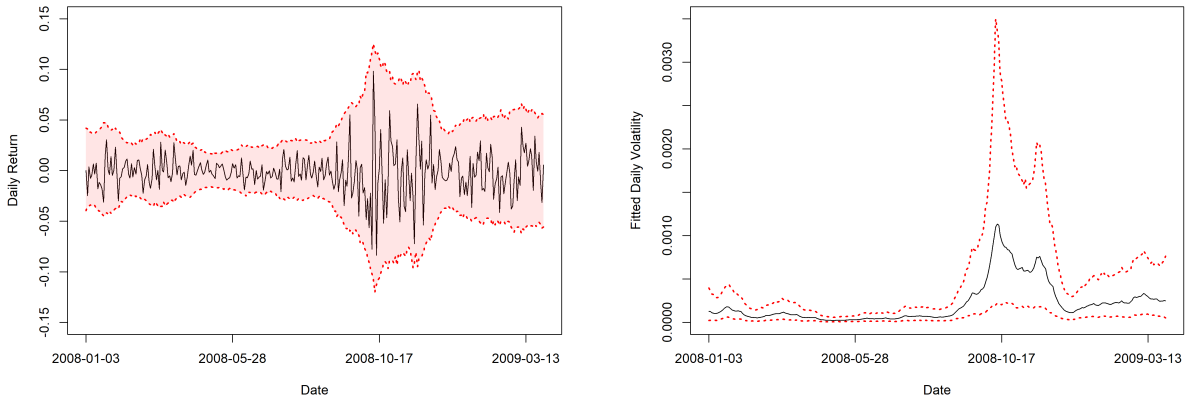


Figure 3.2: The left panel presents the daily returns of the S&P 500 index from January 2008 to March 2009. The large fluctuations around October 2008 indicate the climax of the global financial crisis. The 2.5% and 97.5% quantiles of the sampled returns are shown with red dashed lines and the corresponding 95% credible interval for the returns is highlighted in red. The right panel presents the fitted daily volatility (black line) with a 95% credible interval (red dashed lines); the high volatility around the climax is well-captured.

produces more accurate parameter estimates with the help of its weight tempering strategy, as demonstrated in the simulation study.

Our sampler can be adapted for broader use with different models and setups. First, if there are any model parameters without closed-form conditional distributions (e.g., ϵ_t chosen to be a t-distribution and thus no conjugacy available for σ_h^2), or if the hyperparameters (α, β) of the stable distribution also need to be estimated, this can be handled by incorporating an additional block of Metropolis updates within the particle Gibbs algorithm. However, the tuning of Metropolis kernels needs to be done carefully (Vankov et al., 2019). Second, the computation of the tempered weights for cAPF can be adapted as appropriate to cover the high-density regions of the true importance weights. For example, if the likelihood involves a stable distribution with $0.5 < \alpha < 1$, the Lévy distribution (which corresponds to a stable distribution with $\alpha = 0.5$) can be a better choice of approximating distribution than a Cauchy. However, for a stable distribution with $\alpha < 0.5$ or other kinds of intractable distributions, alternative approximation schemes or numerical methods might be necessary to perform weight tempering efficiently. Finally, other application areas that use SSMs with intractable likelihoods could be considered, e.g., stochastic

kinetic models in systems biology (Owen et al., 2015; Lowe et al., 2023).

CV	ϕ	Algorithm	RMSE τ	RMSE ϕ	RMSE σ_h^2
10	0.9	ABC-PG-cBF	0.518	0.068	0.463
		ABC-PG-cBFAS	0.594	0.078	0.505
		ABC-PG-cAPF	0.173	0.025	0.141
10	0.95	ABC-PG-cBF	0.647	0.086	0.420
		ABC-PG-cBFAS	0.747	0.099	0.489
		ABC-PG-cAPF	0.324	0.049	0.145
10	0.98	ABC-PG-cBF	0.817	0.112	0.414
		ABC-PG-cBFAS	0.916	0.131	0.510
		ABC-PG-cAPF	0.547	0.079	0.214
1	0.9	ABC-PG-cBF	0.571	0.080	0.441
		ABC-PG-cBFAS	0.663	0.094	0.486
		ABC-PG-cAPF	0.157	0.029	0.179
1	0.95	ABC-PG-cBF	0.819	0.116	0.466
		ABC-PG-cBFAS	0.946	0.134	0.524
		ABC-PG-cAPF	0.425	0.068	0.216
1	0.98	ABC-PG-cBF	0.988	0.142	0.449
		ABC-PG-cBFAS	1.125	0.161	0.516
		ABC-PG-cAPF	0.645	0.099	0.245
0.1	0.9	ABC-PG-cBF	0.578	0.084	0.455
		ABC-PG-cBFAS	0.709	0.103	0.520
		ABC-PG-cAPF	0.171	0.033	0.254
0.1	0.95	ABC-PG-cBF	0.855	0.123	0.463
		ABC-PG-cBFAS	1.012	0.146	0.526
		ABC-PG-cAPF	0.484	0.078	0.259
0.1	0.98	ABC-PG-cBF	1.038	0.150	0.459
		ABC-PG-cBFAS	1.214	0.176	0.534
		ABC-PG-cAPF	0.699	0.109	0.268

Table 3.1: RMSEs of the SVM parameter estimates for the three particle Gibbs algorithms, based on 100 simulated datasets with $T = 100$, $N = 100$, $Z_t \sim SD(1.75, 0.1, 1, 0)$, $\epsilon = 0.001$.

CV	ϕ	Algorithm	RMSE τ	RMSE ϕ	RMSE σ_h^2
10	0.9	ABC-PG-cBF	0.514	0.067	0.455
		ABC-PG-cBFAS	0.610	0.079	0.503
		ABC-PG-cAPF	0.170	0.025	0.144
10	0.95	ABC-PG-cBF	0.653	0.087	0.431
		ABC-PG-cBFAS	0.768	0.102	0.502
		ABC-PG-cAPF	0.335	0.051	0.156
10	0.98	ABC-PG-cBF	0.879	0.120	0.475
		ABC-PG-cBFAS	0.993	0.136	0.530
		ABC-PG-cAPF	0.557	0.080	0.218
1	0.9	ABC-PG-cBF	0.564	0.080	0.435
		ABC-PG-cBFAS	0.686	0.098	0.503
		ABC-PG-cAPF	0.148	0.029	0.178
1	0.95	ABC-PG-cBF	0.831	0.117	0.477
		ABC-PG-cBFAS	0.959	0.136	0.531
		ABC-PG-cAPF	0.443	0.071	0.226
1	0.98	ABC-PG-cBF	1.025	0.148	0.472
		ABC-PG-cBFAS	1.167	0.167	0.550
		ABC-PG-cAPF	0.677	0.104	0.265
0.1	0.9	ABC-PG-cBF	0.622	0.089	0.495
		ABC-PG-cBFAS	0.745	0.109	0.548
		ABC-PG-cAPF	0.200	0.037	0.265
0.1	0.95	ABC-PG-cBF	0.892	0.129	0.487
		ABC-PG-cBFAS	1.041	0.151	0.550
		ABC-PG-cAPF	0.482	0.078	0.261
0.1	0.98	ABC-PG-cBF	1.101	0.160	0.504
		ABC-PG-cBFAS	1.246	0.181	0.560
		ABC-PG-cAPF	0.707	0.110	0.274

Table 3.2: RMSEs of the SVM parameter estimates for the three particle Gibbs algorithms, based on 100 simulated datasets with $T = 100$, $N = 100$, $Z_t \sim SD(1.7, 0.3, 1, 0)$, $\epsilon = 0.001$.

CV	ϕ	Algorithm	RMSE τ	RMSE ϕ	RMSE σ_h^2
10	0.9	ABC-PG-cBF	0.681	0.088	0.614
		ABC-PG-cBFAS	0.812	0.105	0.722
		ABC-PG-cAPF	0.174	0.027	0.158
10	0.95	ABC-PG-cBF	0.830	0.110	0.596
		ABC-PG-cBFAS	0.986	0.130	0.702
		ABC-PG-cAPF	0.378	0.057	0.179
10	0.98	ABC-PG-cBF	1.007	0.137	0.566
		ABC-PG-cBFAS	1.174	0.158	0.685
		ABC-PG-cAPF	0.615	0.090	0.253
1	0.9	ABC-PG-cBF	0.714	0.101	0.569
		ABC-PG-cBFAS	0.889	0.126	0.671
		ABC-PG-cAPF	0.174	0.035	0.202
1	0.95	ABC-PG-cBF	0.960	0.136	0.580
		ABC-PG-cBFAS	1.153	0.163	0.684
		ABC-PG-cAPF	0.482	0.078	0.254
1	0.98	ABC-PG-cBF	1.145	0.164	0.579
		ABC-PG-cBFAS	1.338	0.192	0.681
		ABC-PG-cAPF	0.708	0.111	0.290
0.1	0.9	ABC-PG-cBF	0.739	0.108	0.587
		ABC-PG-cBFAS	0.896	0.131	0.674
		ABC-PG-cAPF	0.204	0.041	0.286
0.1	0.95	ABC-PG-cBF	1.018	0.148	0.583
		ABC-PG-cBFAS	1.209	0.176	0.682
		ABC-PG-cAPF	0.553	0.091	0.306
0.1	0.98	ABC-PG-cBF	1.210	0.176	0.591
		ABC-PG-cBFAS	1.392	0.203	0.671
		ABC-PG-cAPF	0.754	0.120	0.309

Table 3.3: RMSEs of the SVM parameter estimates for the three particle Gibbs algorithms, based on 100 simulated datasets with $T = 100$, $N = 100$, $Z_t \sim SD(1.5, -0.3, 1, 0)$, $\epsilon = 0.001$.

Parameter	Estimate	95% credible interval
τ	-0.294	(-0.639, -0.042)
ϕ	0.967	(0.930, 0.995)
σ_h^2	0.098	(0.052, 0.174)

Table 3.4: Estimates and credible intervals of τ, ϕ and σ_h^2 based on fitting an SVM to S&P 500 Index data from January 2008 to March 2009. The numerical results were obtained from the ABC-PG-cAPF algorithm with $N = 500$, $Z_t \sim SD(1.725, 0.0915, 1, 0)$, $\epsilon = 0.001$, and 5000 posterior samples.

Chapter 4

Inference on a Markovian Compartmental Model for COVID-19 in Ontario

4.1 Introduction to compartmental models in epidemiology

The outbreak of COVID-19 caused by SARS-CoV-2 since the end of 2019 in Wuhan, China has deeply impacted human society (Chen et al., 2020). To effectively control the spread of the disease, national and local governments have made different public policies restricting the contact of the communities from time to time. The stringency of the policy has been studied and proved its effectiveness (Tian et al., 2021). Meantime, as a result of the evolvement of the virus, different variants have also been detected (Harvey et al., 2021; Choi and Smith, 2021; Burki, 2021); the later-detected variant could infect those individuals who have been infected previously, and re-detectable positive cases are observed in practice (He and Dong, 2022). Compartment models are a common class of models used in epidemiological studies and thus have been widely applied to model the spread of COVID-19 (Roberto Telles et al., 2021; Keller et al., 2022; Kim et al., 2023; Murphy et al., 2023; Zhao and Wong, 2023). Given the data, e.g., variants and policy stringency, that would contribute to the spread of the disease, it is natural to consider incorporating those data in the compartmental models to improve the accuracy and mimic the realistic situation (Liu et al., 2020; Sartorius et al., 2021) by reducing the improper assumptions that are commonly made in epidemiological studies (Ioannidis et al., 2022).

Generally, the observed data is reported daily or weekly by public health departments for infectious diseases. For example, the spread of COVID-19 in Ontario is investigated in this project, and the observed data is reported by Public Health Ontario daily. Thus we assume a sequence of the time $k = 1, \ldots, K$ to denote the observation time. A typical compartmental model used to model the data divides the population (denoted by N) into three subgroups: the proportion of the susceptible individuals who can potentially get infected after being exposed to the target disease (denoted by S), the proportion of the infected individuals who are confirmed with the disease (denoted by I) and the proportion of the recovered individuals who have been recovered from the disease (denoted by R), with $S + I + R = 1$. The three compartments vary over time depending on some transmission periods τ's and characterize a dynamic system via ordinary differential equations (ODEs)

$$\begin{aligned}
\frac{dS}{dt} &= -\frac{SI}{\tau_{SI}} \\
\frac{dI}{dt} &= \frac{SI}{\tau_{SI}} - \frac{I}{\tau_{IR}} \\
\frac{dR}{dt} &= \frac{I}{\tau_{IR}}
\end{aligned}$$

and the observed count of the infected individuals $N \times I_k^{obs}$ are assigned a likelihood distribution (commonly a negative binomial distribution) with $E[I_k^{obs}]$ equal to the ODE solution at time k to link the observations and the ODE system. Moreover, there are some extensions of the presented compartmental models that divide the population into more subgroups (Lekone and Finkenstädt, 2006; Zhao and Wong, 2023), such as the exposed individuals (denoted by E), the hospitalized individuals (denoted by H), the dead individuals (denoted by D), etc. All of the extended models follow the same setup as introduced. Approaches to the ODE systems usually solve the system in the following steps: (1) take an initial condition and parameters, (2) run a numerical solver to obtain a deterministic I_{ODE}, and (3) use a negative binomial distribution to compute the likelihood of the observed $N \times I_k^{obs}$ with the solved I_{ODE} at time k as the mean. The ODE system provides a continuous time index for all compartments and can be conveniently solved using ODE solvers; however, it only allows constant or time-varying deterministic parameters so no stochastic trajectories can be generated (given a set of initial conditions and parameter values). With the ODE formulation, the likelihood has to account for all the uncertainty in the noisy observations.

Another common way of handling the system is to discretize time by converting the differential equations to difference equations (Lekone and Finkenstädt, 2006; Liu et al.,

2020); thus the model can be reformulated as

$$S_{t+1} = S_t - \frac{S_t I_t}{\tau_{SI}}$$
$$I_{t+1} = I_t + \frac{S_t I_t}{\tau_{SI}} - \frac{I_t}{\tau_{IR}}$$
$$R_{t+1} = R_t + \frac{I_t}{\tau_{IR}}$$

Now the model has the same structure as the GARCH class models where (S_t, I_t, R_t) represents the hidden deterministic process with observed I_t^{obs} conditional on it at each t, so there are tools that can be adapted for solving it. Compared to the ODE system, the difference equations can only consider a fixed time difference between two successively observed data points (e.g., daily data or monthly data) and can naturally be set as the time interval of the observed data; for simplicity, we use the same interval for t and k and $t = k$ for all discrete-time compartmental models in the following context to simplify the notations. This discrete-time system can suffer from handling the data where the observation time is not equally distanced (e.g., observed randomly in time); however, this form allows us to adopt stochastic parameters and thus generate stochastic trajectories.

The importance of the stochasticity of the compartmental model has been discussed (Ball et al., 1997; Lekone and Finkenstädt, 2006). The stochasticity potentially allows the trajectory to evolve in different directions abruptly; with a stochastic involvement of a trajectory, the uncertainty can be explained by the randomness in the system itself instead of introducing an extra likelihood distribution for the observations. Lekone and Finkenstädt (2006) present a stochastic compartmental model for Ebola for which the stochasticity of the transmission is not neglectable. However, they did not consider incorporating any covariates to model the stochastic transmission of the compartments over time. In practice, the transmission periods τ's are also often assumed time-varying to incorporate some observed covariates (e.g., mobility and stringency of the public policy), but can only be assumed as a function of those covariates, rather than being stochastic, to be compatible with an ODE system. For the difference equation system, Liu et al. (2020) applied generalized linear models (GLMs) to model τ's to incorporate the Google mobility data for COVID-19 in the compartmental model; however, they only build up a short-term discrete-time model which still assigns a negative binomial likelihood to the observations. They utilized the adaptive sequential Monte Carlo and Hamiltonian Monte Carlo kernel for the tractable posterior in their setting, which inspired us to propose a sampling method that incorporates SMC, albeit in a different setting.

In this paper, we propose a compartmental model where τ's are modeled using GLMs

with the covariates including the mobility, the stringency of the public policy, the variants of SARS-CoV-2, with noise terms embedded within the GLMs at each t, and thus the compartment transmission is stochastic. We also extend the common compartmental model by allowing recovered individuals to return to the susceptible group after a period of antibody decay and/or a new variant starts dominating in the population. As a result, the process of the compartments is now a Markov process and we no longer need an extra level of likelihood for I^{obs} given I to explain the uncertainty in the observations. However, the model comes with a computational difficulty: the transition probability of the Markov process, i.e., $p((S_{t+1}, I_{t+1}, R_{t+1}) \mid (S_t, I_t, R_t))$, is intractable. Therefore, the current approaches for the common ODE-based compartmental models are inapplicable.

A natural way of handling intractable probabilities is via approximate Bayesian computation (ABC) techniques, and there are a few approaches adopting ABC techniques that one can consider. For example, ABC-SMC methods are originally designed for this online problem; however, it suffers from particle degeneracy and we have to start with a very large particle size to have some initial starting points "close" enough to the true values, and thus it is expensive for high-dimensional problems such as the compartmental models. As an alternative, ABC-PMCMC avoid the degeneracy by stochastically probing new particles, but when using a sharp ABC kernel (to have an accurate ABC approximation), it is easy to get stuck in a local mode. Therefore, we propose to apply an ABC-based SMC^2 algorithm (Chopin et al., 2013; Kerama et al., 2022), named ABC-SMC^2 in our context. We construct an outer SMC for all τ's which adopts (i) an inner ABC-based bootstrap filter (Gordon et al., 1993; Marin et al., 2012) for estimating the intractable marginal likelihood and (ii) a particle Markov chain Monte Carlo (PMCMC, Andrieu et al., 2010) kernel to stochastically update the parameters to avoid particle decay after resampling. We shall show that ABC-SMC^2 is an effective approach to handle the proposed model with intractable transmission probability.

The remainder of the paper is laid out as follows. In Section 4.2, we illustrate the construction of our proposed compartmental models and how to incorporate the covariates using GLMs; we also present the proposed ABC-SMC^2 algorithm. In Section 4.3, we implement a simulation study that displays the fitting performances of the model using ABC-SMC^2.

4.2 Methodology

4.2.1 Model setup

We assume four compartments S, E, I and R in our compartmental model with the difference equations

$$\begin{aligned} S_{t+1} &= S_t + \frac{R_t}{\tau_{RS}(t)} - \frac{S_t I_t}{\tau_{SE}(t)} \\ E_{t+1} &= E_t + \frac{S_t I_t}{\tau_{SI}(t)} - \frac{E_t}{\tau_{EI}(t)} \\ I_{t+1} &= I_t + \frac{E_t}{\tau_{EI}(t)} - \frac{I_t}{\tau_{IR}(t)} \\ R_{t+1} &= R_t + \frac{I_t}{\tau_{IR}(t)} - \frac{R_t}{\tau_{RS}(t)} \end{aligned} \tag{4.1}$$

with S_0, E_0, R_0 and I_t for all t observed. Given the observed time-series covariates including the mobility and stringency (denoted by $X_{ms}(t)$), wastewater (denoted by $W(t)$), variants of the the virus (denoted by $Y(t)$), the time-varying transmission periods τ's are assumed to follow a generalized linear regression with coefficients β's and variance σ^2's as

$$\begin{aligned} \tau_{SE}(t) &\sim LogNormal(\beta_{SE}^{\mathrm{T}} X_{ms}(t), \sigma_{SE}^2) \\ \tau_{EI}(t) &\sim LogNormal(\beta_{EI} W(t), \sigma_{EI}^2) \\ \tau_{IR}(t) &\sim LogNormal(\beta_{IR}^{\mathrm{T}} Y(t), \sigma_{IR}^2) \\ \tau_{RS}(t) &\sim LogNormal(\beta_{RS}^{\mathrm{T}} Y(t), \sigma_{RS}^2) \end{aligned}$$

with $\beta \sim N(0, \sigma_\beta^2)$ for all β's and $\sigma \sim \Gamma(k_\Gamma, \theta_\Gamma)$, with k_Γ for shape and θ_Γ for shape, for all σ's as the priors. Furthermore, we define notations

$$\begin{aligned} \theta &= (\beta_{SE}, \sigma_{SE}, \beta_{EI}, \sigma_{EI}, \beta_{IR}, \sigma_{IR}, \beta_{RS}, \sigma_{RS}) \\ U_t &= (S_t, E_t, I_t, R_t) \\ V_t &= (1/\tau_{SE}(t), 1/\tau_{EI}(t), 1/\tau_{IR}(t), 1/\tau_{RS}(t)) \end{aligned}$$

then U_t represents a Markov process conditional on θ and $X_{1:T-1}$ as

$$U_{t+1} = U_t + \begin{bmatrix} -S_t I_t & 0 & 0 & R_t \\ S_t I_t & -E_t & 0 & 0 \\ 0 & E_t & -I_t & 0 \\ 0 & 0 & I_t & -R_t \end{bmatrix} \begin{bmatrix} \frac{1}{\tau_{SE}(t)} \\ \frac{1}{\tau_{EI}(t)} \\ \frac{1}{\tau_{IR}(t)} \\ \frac{1}{\tau_{RS}(t)} \end{bmatrix} := U_t + \mathbf{M}_t V_t$$

which illustrates that U_{t+1} is a linear transformation of V_t given U_t (denoted by $U_{t+1} = \mathbf{F}(U_t, V_t)$). Then the transition density of the Markov process satisfies

$$p(U_{t+1} \mid U_t, X_t, \theta) = \int p(U_{t+1}, V_t \mid U_t, X_t, \theta) dV_t = \int \mathbb{1}_{\{V_t: \mathbf{F}(U_t, V_t) = U_{t+1}\}} p(V_t \mid X_t, \theta) dV_t.$$

However, the integral involves a linear combination of independent lognormal variables that has no analytic form; as a result, the Markov process has an intractable density $p(U_{t+1} \mid U_t, X_t, \theta)$. Moreover, as the model incorporates the observations $\{I_t^{obs}\}_{t=1}^T$ as the I compartment, the observation probability $p(I_{t+1}^{obs} \mid U_t, X_t, \theta)$ is also intractable and cannot be directly employed to fit the observation data.

To handle such intractable transition densities, we shall adopt the ABC methods by introducing auxiliary variables U_t^*. For each θ, we can generate the auxiliary $U_{1:T}^*$ (let $U_0^* = U_0$ for convenience) sequentially from $p(U_{t+1} \mid U_t, X_t, \theta)$ over t and assume the observations $I_{1:T}$ are conditional on $U_{1:T}^*$ via an ABC kernel. Then the extended model can be demonstrated as:

$$\begin{array}{ccccccc} U_0 & \xrightarrow{\theta, X_0} & U_1^* & \xrightarrow{\theta, X_1} & \cdots & \xrightarrow{\theta, X_{T-1}} & U_T^* \\ & & \downarrow & & & & \downarrow \\ & & I_1 & & & & I_T \end{array}$$

which adopts an SSM structure with intractable transition probabilities.

4.2.2 The ABC-SMC^2 method

The SMC^2 method was originally proposed by Chopin et al. (2013) for SSMs and consists of an outer SMC for the unknown parameters, an inner SMC for the hidden states, and a PMCMC kernel that incorporates the inner SMC for stochastically updating parameters. In our compartmental model setup, due to the intractability of the transition probability $p(U_{t+1} \mid U_t, X_t, \theta)$, the inner SMC cannot be directly implemented, so the original SMC^2 method is not implementable either. To overcome the problem, we shall build up an inner ABC-SMC to bypass the evaluation of the intractable probability for $U_{1:T}^*$ given θ; a PMCMC kernel for stochastically updating θ that incorporates the inner ABC-SMC is also built as a byproduct, named ABC-PMCMC; the outer SMC for θ, the inner ABC-SMC, and the ABC-PMCMC compose the ABC-SMC^2 method that we use in this paper. The ABC-SMC^2 method has been discussed by Kerama et al. (2022); they assume a sequence of $\infty > \epsilon_1 > \cdots > \epsilon_T$ for the ABC kernel to build up a sequence of distributions that are "closer" to the true target over iterations. However, they do not consider the observations

Algorithm 7: ABC Bootstrap Filter

input: initial states U_0, observations $I_{1:T}$, covariates $X_{0:T-1}$, parameters θ, transition function $\mathbf{F}$, ABC kernel K_ϵ;

draw $V_0^{*(n)} \sim p(V_0 \mid \theta, X_0)$ and generate $U_1^{*(n)} = \mathbf{F}(U_0, V_0^{*(n)})$ for $n = 1, \ldots, N$;

$w_1^{(n)} = K_\epsilon(I_1 \mid U_1^{*(n)})$ for $n = 1, \ldots, N$;

for *t in 2:T* **do**

  resample $\{{U_{t-1}^*}^{(n)}\}_{n=1}^N$ from $\{({U_{t-1}^*}^{(n)}, w_{t-1}^{(n)})\}_{n=1}^N$;

  for *n in 1:N* **do**

    draw ${V_{t-1}^*}^{(n)} \sim p(V_{t-1} \mid \theta, X_{t-1})$ and generate $U_t^{*(n)} = \mathbf{F}({U_{t-1}^*}^{(n)}, {V_{t-1}^*}^{(n)})$;

    $w_t^{(n)} = K_\epsilon(I_t \mid U_t^{*(n)})$;

  end

end

return $\log(\hat{p}_\epsilon(I_{1:T} \mid \theta)) = \sum_{t=1}^T \log(\sum_{n=1}^N w_t^{(n)}/N)$;

to be time-series data, and thus their ABC-SMC^2 method is not designed for online analysis. In contrast, as the observations sequentially arrive in our model assumption, we shall present our ABC-SMC^2 for online inference. In the following, we shall discuss each of the three components of our ABC-SMC^2 in detail.

Model 4.1 implies that U_{t+1} is a mapping of U_t and V_t (i.e., $U_{t+1} = \mathbf{F}(U_t, V_t)$ for a function $\mathbf{F}$). We propose to implement an ABC-based bootstrap filter in Algorithm 7 as the inner SMC to estimate the marginal propagation likelihood: for each t, sample (V_t^*, θ) from $p(V_t \mid \theta, X_t)p(\theta)$, then compute the mapping of (U_t^*, V_t^*) as $U_{t+1}^* = \mathbf{F}(U_t^*, V_t^*)$ and last compute the ABC kernel density $K_\epsilon(I_{t+1} \mid U_{t+1}^*) = K_\epsilon(I_{t+1} \mid I_{t+1}^*)$; with the generated particle weights $\{w_t^{(n)}\}_{n=1}^N$ at step t, the estimated marginal propagation likelihood $\hat{p}_\epsilon(I_t \mid I_{t-1}, \theta) = \frac{1}{N}\sum_{n=1}^N w_t^{(n)}$ (with $\hat{p}_\epsilon(I_1 \mid I_0, \theta) = \hat{p}_\epsilon(I_1 \mid \theta)$).

Following Chopin et al. (2013), we use a particle Metropolis-Hastings kernel presented in Algorithm 8. The kernel proposes a new candidate of θ from a propagation kernel H and evaluates the estimated likelihood as $\hat{p}_\epsilon(I_{1:T} \mid \theta) = \prod_{t=1}^T \hat{p}_\epsilon(I_t \mid I_{t-1}, \theta)$, which will be used to compute the Metropolis-Hastings acceptance rate to determine whether to accept the new candidate.

After presenting the first two algorithms, we can now construct an outer SMC for θ incorporating the previous two algorithms to build up an ABC-SMC^2 algorithm, which is summarized in Algorithm 9. Given a prechosen particle size M for the outer SMC, we start with generating M initial θ's from the prior; then every time a new observation I_t^*

Algorithm 8: ABC Particle Metropolis-Hastings Kernel

input: Prior $p(\theta)$, propagation kernel H, initial states U_0, observations $I_{1:T}$, covariates $X_{0:T-1}$, initial parameters θ_c, transition function $\mathbf{F}$, ABC kernel K_ϵ;
$p_c = \log(p(\theta_c))$;
$l_c =$ run Algorithm 7 with θ_c;
propagate new parameters $\theta_n \sim H(\theta \mid \theta_c)$;
$p_n = \log(p(\theta_n))$;
$l_n =$ run Algorithm 7 with θ_n;
draw $u \sim U(0, 1)$;
if $u < \exp[l_n - l_c + p_n - p_c + \log\{H(\theta_c \mid \theta_n)\} - \log\{H(\theta_n \mid \theta_c)\}]$ **then**
  return θ_n;
end
else
  return θ_c;
end

arrives, we do the following steps:

1. update the particle weight of each of the M θ's using the estimated probability $\hat{p}_\epsilon(I_t \mid I_{t-1}, \theta)$ via the inner ABC-SMC (i.e., Algorithm 7) at time t;

2. compute the effective sample size (ESS) using the updated particle weights. If ESS is greater than a prechosen threshold, go to step 1 of $t + 1$; otherwise go to step 3 and 4;

3. resample the θ's according to the current particle weights;

4. stochastically update each resampled θ using the PMCMC kernel (i.e., Algorithm 8) up to time t.

There are two choices for the propagation kernel H of Algorithm 8 in Algorithm 9 suggested by Chopin et al. (2013): given the particles $\{\theta_c^{(m)}\}_{m=1}^M$, to obtain the updated particle of each $\theta_c^{(m)}$, denoted by $\theta_n^{(m)}$, one can use (i) a random-walk kernel $H_{rw}(\theta_n \mid \theta_c^{(m)})$, e.g., $\theta_n^{(m)} \sim N\left(\theta_c^{(m)}, c\widehat{\Sigma}_{\theta_c,M}\right)$ where $\widehat{\Sigma}_{\theta_c,M}$ indicates the empirical covariance matrix computed with $\{\theta_c^{(m)}\}_{m=1}^M$ and c is a scaling constant; (ii) an independent kernel $H_{ind}(\theta_n)$, e.g., $\theta_n^{(m)} \sim N\left(\frac{1}{M}\sum_{m=1}^M \theta_c^{(m)}, \widehat{\Sigma}_{\theta_c,M}\right)$. In particular for σ's, since they must be positive, one can take

Algorithm 9: ABC SMC^2 Algorithm

input: prior $p(\theta)$, outer particle size M, inner particle size N, observations $I_{1:T}$, threshold γ;
sample $\theta^{(m)} \sim p(\theta)$ with $w^{(m)} = 1$ for $m = 1, \ldots, M$;
for *t in 1:T* **do**
  for *m in 1:M* **do**
    run step t of Algorithm 7 with $\theta^{(m)}$ and compute $\hat{p}_\epsilon(I_t \mid I_{t-1}, \theta^{(m)}) = \frac{1}{N}\sum_{n=1}^{N} w_t^{(n)}$;
    update $w^{(m)} = w^{(m)} \times \hat{p}_\epsilon(I_t \mid I_{t-1}, \theta^{(m)})$;
  end
  if $\left(\sum w^{(m)}\right)^2 / \sum \left(w^{(m)}\right)^2 < \gamma M$ **then**
    for *m in 1:M* **do**
      resample $\tilde{\theta}^{(m)}$ from $\{(\theta^{(m)}, w^{(m)})\}_{m=1}^{M}$;
      $\theta^{(m)}$ = updated $\tilde{\theta}^{(m)}$ via Algorithm 8 with $w^{(m)} = 1$;
    end
  end
end
return $\theta^{(1:M)}$;

the absolute values of the samples, or keep sampling until obtaining positive values, or use a distribution with positive support such as a Gamma distribution, with the Metropolis-Hastings acceptance rate computed accordingly.

In summary, the introduced ABC-SMC^2 algorithm has the following features: it adopts an ABC-SMC as the inner SMC to handle the intractable likelihood. The resampling step in Step 3 is implemented when ESS is low (i.e., weight degeneracy is severe) and the PMCMC kernel in Step 4 allows the resampled θ's to be stochastically updated for a higher diversity of the particles (i.e., to reduce path degeneracy).

4.3 Simulation study

In this section, we begin with a short-term window from February 15, 2020, to October 31, 2020, and obtain the mobility data from Google (Google LLC, 2022) and the stringency level of the public policy from the Bank of Canada (Bank of Canada, 2022). In particular, the mobility data consists of a table of mobility change for different locations, and we take

the daily average over all locations and scale the mobility values into the range of $(0, 1)$, denoted by $X_m(t)$; we also scale the stringency indices into the range of $(0, 1)$, denoted by $X_s(t)$. Moreover, during this period, wastewater data was not collected due to the lack of realization of its significance, and no dominant variants were distinguished. Due to the short time window, we also tentatively omit the transition from R to S to simplify the model. Therefore, the GLMs are characterized as

$$\begin{aligned}
\tau_{SE}(t) &\sim LogNormal(\beta_{0,SE} + \beta_{1,SE}X_m(t) + \beta_{2,SE}X_s(t), \sigma^2_{SE}) \\
\tau_{EI}(t) &\sim LogNormal(\beta_{EI}, \sigma^2_{EI}) \\
\tau_{IR}(t) &\sim LogNormal(\beta_{IR}, \sigma^2_{IR})
\end{aligned}$$

and the parameter vector $\theta = (\beta_{0,SE}, \beta_{1,SE}, \beta_{2,SE}, \sigma_{SE}, \beta_{EI}, \sigma_{EI}, \beta_{IR}, \sigma_{IR})$ shall be estimated using ABC-SMC^2. To further simplify the simulation settings, we assume that all elements of U_t are observable for all t; then for each auxiliary variable U_t^* t, the ABC-kernel K_ϵ is computed as $N(||U_t - U_t^*||; \mu = 0, \sigma = \epsilon)$ where $||\cdot||$ denotes the l^2-norm.

We start the simulation by setting the true parameters $\theta = (0.5, 0.5, 0.5, 0.01, 1, 0.01, 2, 0.01)$. In ABC-SMC^2, we assign the following prior

$$\begin{aligned}
\beta_{0,SE} &\sim N(0, 1), \quad \beta_{1,SE} \sim N(0, 1), \\
\beta_{2,SE} &\sim N(0, 1), \quad \sigma_{SE} \sim \Gamma(0.2, 0.5), \\
\beta_{EI} &\sim N(0, 1), \quad \sigma_{EI} \sim \Gamma(0.2, 0.5), \\
\beta_{IR} &\sim N(0, 1), \quad \sigma_{IR} \sim \Gamma(0.2, 0.5).
\end{aligned}$$

We also set $\epsilon = 0.001$ for K_ϵ, outer particle size $M = 10000$, inner particle size $N = 100$ and ESS threshold rate $\gamma = 0.5$. To mimic the real population size of Ontario and the spread of COVID-19 in Ontario while keeping the settings as simple as possible, we assume the initial susceptible group is 14 million and there is only 1 initially infected individual with 0 exposure and 0 recovery, i.e., we assume $U_0 = (14000000, 0, 1, 0)$. We chose three types of the propagation kernel H: (i) $H = N\left(\theta_c^{(m)}, \text{diag}(\widehat{\Sigma}_{\theta_c,M})\right)$, (ii) $H = N\left(\frac{1}{M}\sum_{m=1}^{M}\theta_c^{(m)}, \text{diag}(\widehat{\Sigma}_{\theta_c,M})\right)$ for all parameters and keep sampling until σ's are positive, and (iii) H defined in (ii) for all β's and $H = \Gamma(p, q)$ for σ's, where p and q are the shape and scale parameters that satisfies pq equal to the mean of the to-be-updated M σ's and pq^2 equal to their standard deviation. Notably, we ignore the covariance of the parameters for simplicity in this simulation.

Kernel H	$\beta_{0,SE}$	$\beta_{1,SE}$	$\beta_{2,SE}$	σ_{SE}	β_{EI}	σ_{EI}	β_{IR}	σ_{IR}
random walk (i)	0.0372	0.0130	-0.0624	0.0207	-0.0005	0.0047	0.0007	0.0028
	0.1202	0.0434	0.1906	0.0366	0.0128	0.0119	0.0045	0.0100
independent (ii)	0.0372	0.0411	-0.0840	0.0305	0.0031	0.0096	0.0011	0.0072
	0.1102	0.1427	0.1468	0.0488	0.0227	0.0335	0.0062	0.0201
independent (iii)	0.0401	0.0131	-0.0662	0.0038	0.0001	-0.0031	0.0000	-0.0006
	0.0963	0.0655	0.1179	0.0153	0.0040	0.0089	0.0041	0.0124

Table 4.1: A summary of the inference results of ABC-SMC^2 for different propagation kernels H in the PMCMC kernel. The bias (top) and RMSE (bottom) computed with the estimates from the 100 repetitions are reported for each parameter. The true values are set as $\beta_{0,SE} = \beta_{1,SE} = \beta_{2,SE} = 0.5$, $\beta_{EI} = 1$, $\beta_{IR} = 2$ and $\sigma_{SE} = \sigma_{EI} = \sigma_{IR} = 0.01$, and the prior are set as all β's follow $N(0, 1)$ and all σ's follow $\Gamma(0.2, 0.5)$.

For each setting, we implement 100 independent repetitions of implementing ABC-SMC^2; each repetition produces the particle means as the point estimates for the parameters. The numeric results of the simulation are reported in Table 4.1 and visualized in Figure 4.1, 4.2 and 4.3. Overall, the independent kernel H defined in (iii) produced the smallest RMSEs compared to the rest in Table 4.1, and its boxplot presented in Figure 4.3 showed relatively less dispersion. A possible explanation is that the independent Gamma distributions are denser for small values compared to independent Gaussian distributions and thus more easily capture the values around the true values; meanwhile, the Gaussian random walk can easily get stuck in local modes. We also randomly select a set of estimates from the 100 copies and generate the trajectory plots for the four compartments to assess how well the ABC-SMC^2 fits the data, and the plots can be found in Appendix C.1.

4.4 Discussion

In this project, we proposed to incorporate covariate data (including the Google mobility data and the stringency of the public policy) for modeling the transmission periods in the compartmental model using GLMs. The GLM-modelled transmission periods in our model lead to a stochastic system with an intractable transition probability and are assumed to adopt all uncertainty in the system, thus no need for likelihood. This feature distinguishes our work from the existing compartmental models that rely on the assumption of likelihood along with the system, particularly the model proposed by Liu et al. (2020) that also adopts GLMs along with negative likelihood and tractable posterior for Hamiltonian Monte Carlo.

However, this model is computationally difficult to solve, and we proposed to utilize an ABC-SMC^2 method whose performances in the simulation study are promising.

Albeit the relatively good performances of the independent kernel H defined in (iii) in the current simulation, it still produced some outliers that are far from the true values among the 100 repetitions, so the empirical rates of the 95% CIs for some parameters are much lower than 95%. This is the first issue we need to investigate. We assume that all compartments are observed in the simulation, which is infeasible in practice. We shall later change the observation to only the infected individuals to mimic reality and check how our method performs. A longer window such as one-year or two-year data can also be considered. In that case, some facts such as re-detectable positivity, variants of SARS-CoV-2, and wastewater level might be considered. The model can be further extended by introducing more compartments and incorporating vaccination rates e.g., the model considered by Zhao and Wong (2023). However, more compartments come with a higher parameter dimension, which can challenge the computation. Furthermore, an application is needed for the goal of practical application. Public Health Ontario has an open source for the collected COVID-19 data that reports the daily counts of confirmed cases, hospitalizations, deaths, vaccines administered, weekly tests completed, and positivity rates, which can be used for a real-data application.

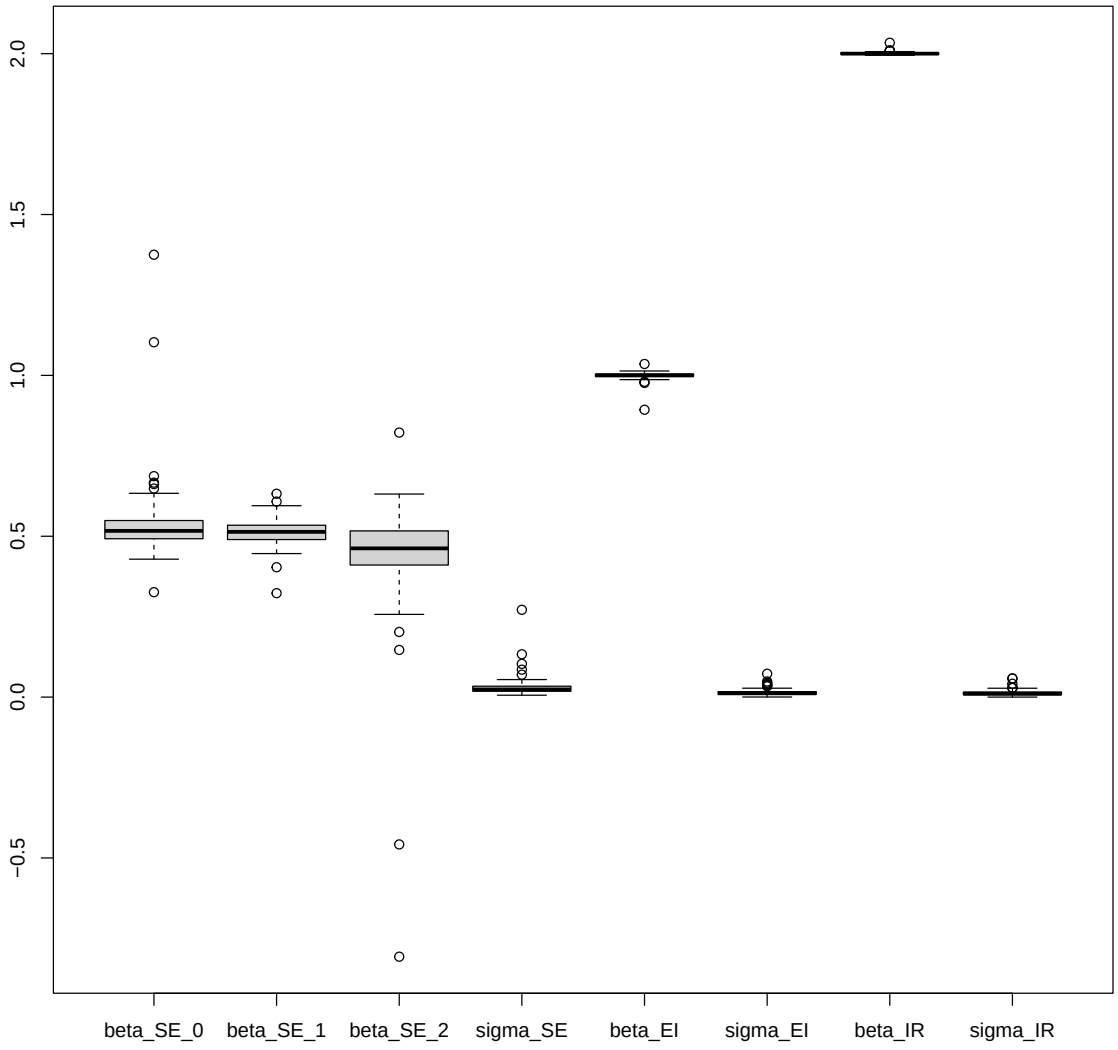


Figure 4.1: The boxplot of the 100 estimates produced from ABC-SMC^2 for each parameter with kernel H defined in (i). The true values are set as $\beta_{0,SE} = \beta_{1,SE} = \beta_{2,SE} = 0.5$, $\beta_{EI} = 1$, $\beta_{IR} = 2$ and $\sigma_{SE} = \sigma_{EI} = \sigma_{IR} = 0.01$, and the prior are set as all β's follow $N(0, 1)$ and all σ's follow $\Gamma(0.2, 0.5)$.

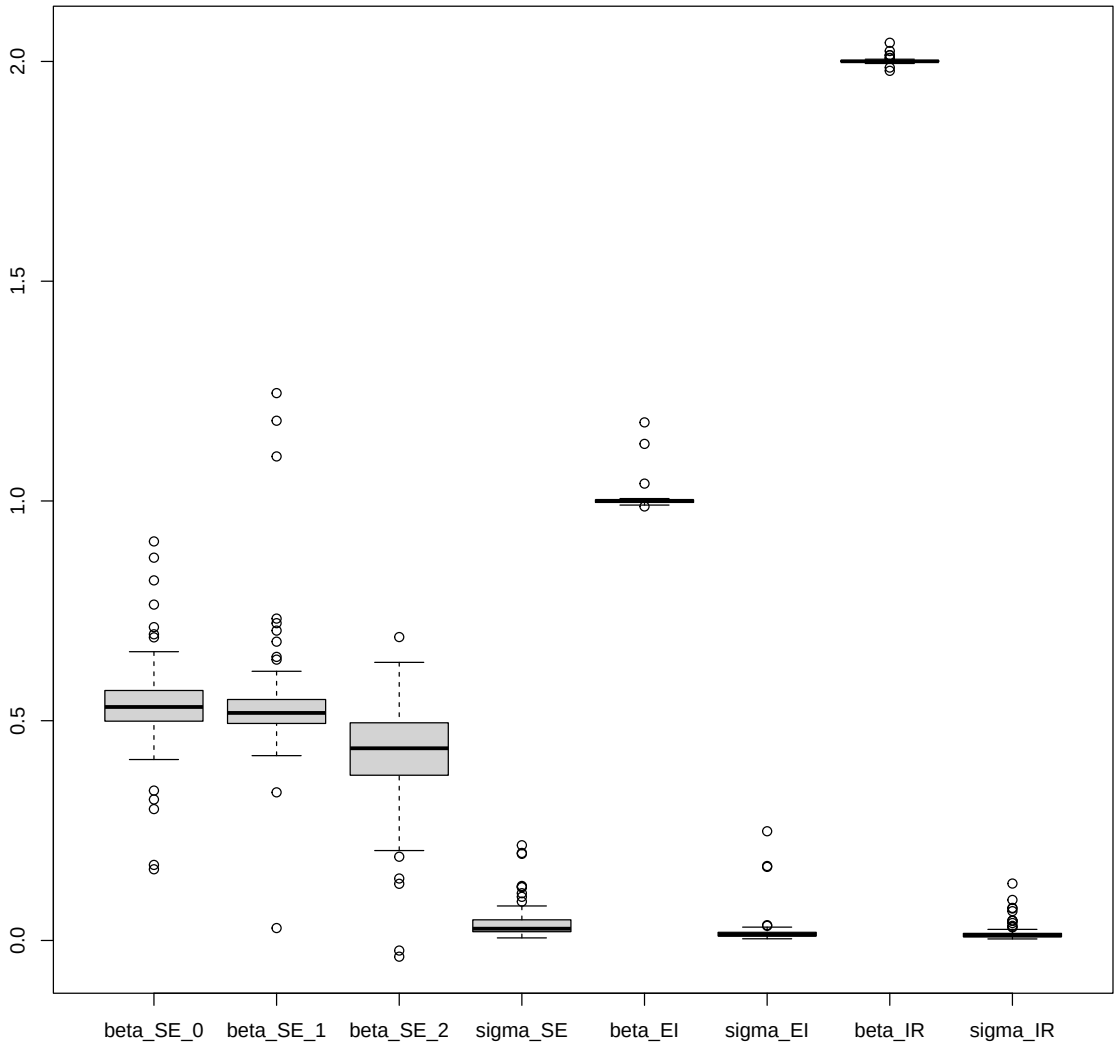


Figure 4.2: The boxplot of the 100 estimates produced from ABC-SMC^2 for each parameter with kernel H defined in (ii). The true values are set as $\beta_{0,SE} = \beta_{1,SE} = \beta_{2,SE} = 0.5$, $\beta_{EI} = 1$, $\beta_{IR} = 2$ and $\sigma_{SE} = \sigma_{EI} = \sigma_{IR} = 0.01$, and the prior are set as all β's follow $N(0, 1)$ and all σ's follow $\Gamma(0.2, 0.5)$.

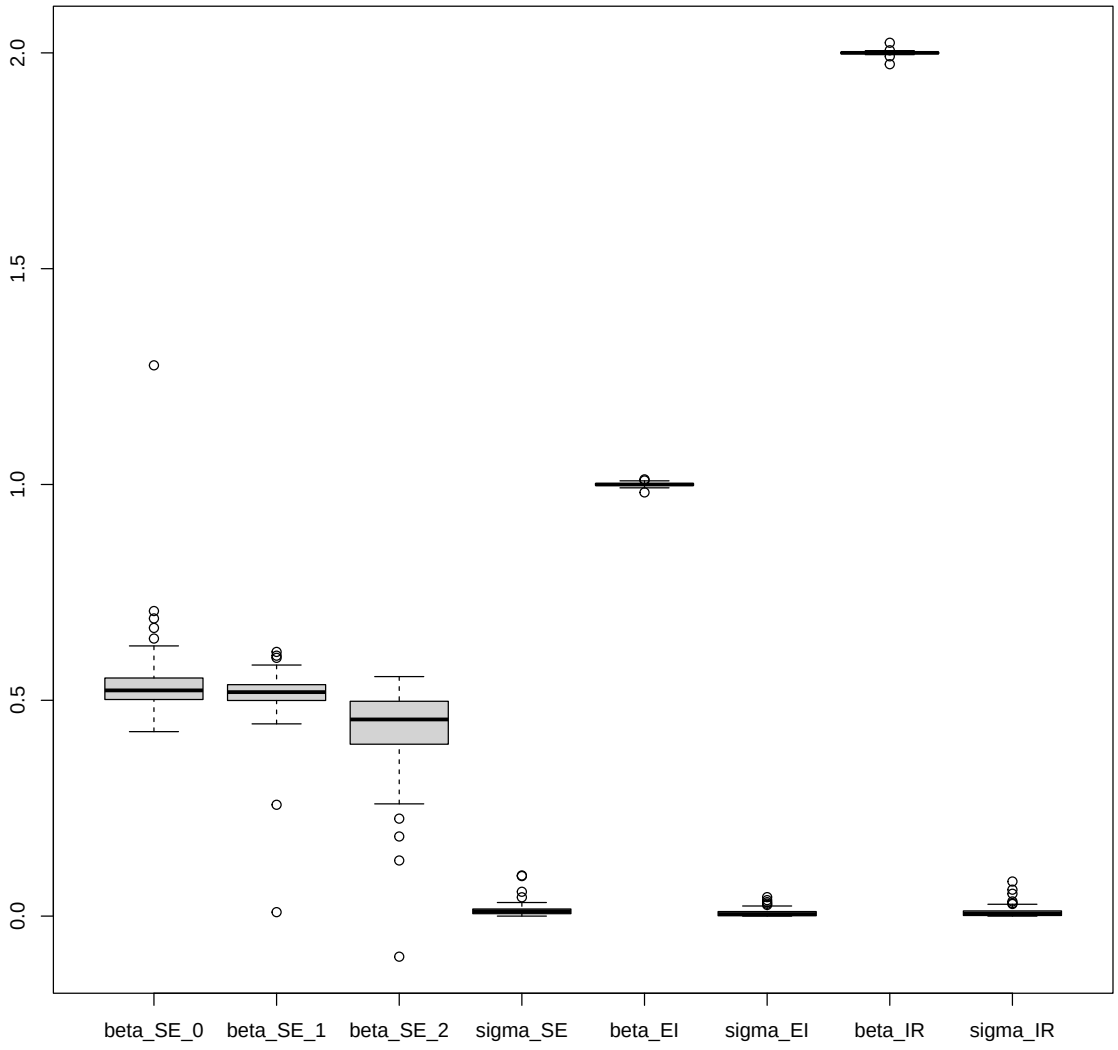


Figure 4.3: The boxplot of the 100 estimates produced from ABC-SMC^2 for each parameter with kernel H defined in (iii). The true values are set as $\beta_{0,SE} = \beta_{1,SE} = \beta_{2,SE} = 0.5$, $\beta_{EI} = 1$, $\beta_{IR} = 2$ and $\sigma_{SE} = \sigma_{EI} = \sigma_{IR} = 0.01$, and the prior are set as all β's follow $N(0, 1)$ and all σ's follow $\Gamma(0.2, 0.5)$.

Chapter 5

Conclusion and Future Work

In this chapter, we present a summary of the thesis and some potential future work for each project.

In Chapter 2, we have introduced a novel SMC method for sampling from general multivariate distributions and how to use it for structural quantity estimation for protein backbones, and the simulation results support the performance of the proposed SMC for solving these problems. In this chapter, we only considered the backbone atoms for the protein problem to simplify the problem. We may consider cases with structural quantity estimation for proteins with side chains or different energy functions as future work.

In Chapter 3, we have proposed a novel conditional SMC method embedded into a particle Gibbs sampler for the stochastic volatility model with Gaussian noise ϵ_t and stable noise Z_t. The simulation studies strongly present the advantages of our method. For future work, we may consider estimating the heavy-tailedness α and the skewness β of the stable noise by adding Metropolis-Hastings update blocks into the particle Gibbs sampler; we can also consider employing other approximation methods for $p(h_{t-1} \mid r_t)$ or assuming ϵ_t following some non-Gaussian distribution such as t-distribution.

In Chapter 4, we propose a different setting for the compartmental model which adopts the stochasticity for the transmission periods by modeling with generalized linear regression incorporating covariates. The novel setting leads to the intractable transit density of the compartment models, to which we applied an ABC-SMC^2 method. We will need to investigate the reason for the low coverage rates of the produced credible intervals in the current experiments and later generalize the simulation settings; we will also implement a real-data analysis to reveal the influence of different covariates on the spread of COVID-19 in Ontario.

References

Adcock, S. A. and J. A. McCammon (2006). Molecular dynamics: survey of methods for simulating the activity of proteins. *Chemical Reviews 106*(5), 1589–1615.

Ali, A. and R. Vijayan (2020). Dynamics of the ace2–sars-cov-2/sars-cov spike protein interface reveal unique mechanisms. *Scientific Reports 10*(1), 1–12.

Andrieu, C. and A. Doucet (2002). Particle filtering for partially observed Gaussian state space models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 64*(4), 827–836.

Andrieu, C., A. Doucet, and R. Holenstein (2010). Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 72*(3), 269–342.

Anfinsen, C. B. (1973). Principles that govern the folding of protein chains. *Science 181*(4096), 223–230.

Ball, F., D. Mollison, and G. Scalia-Tomba (1997). Epidemics with two levels of mixing. *The Annals of Applied Probability*, 46–89.

Bank of Canada (2022). *COVID-19 stringency index*. Ph. D. thesis, https://www.bankofcanada.ca/markets/market-operations-liquidity-provision/covid-19-actions-support-economy-financial-system/covid-19-stringency-index/.

Berman, H. M., J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne (2000). The protein data bank. *Nucleic Acids Research 28*(1), 235–242.

Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics 31*(3), 307–327.

Boltzmann, L. (1868). Studien uber das Gleichgewicht der lebenden Kraft. *Wissenschafiliche Abhandlungen 1*, 49–96.

Bu, Z. and D. J. Callaway (2011). Proteins move! Protein dynamics and long-range allostery in cell signaling. *Advances in Protein Chemistry and Structural Biology 83*, 163–221.

Burki, T. (2021). Understanding variants of sars-cov-2. *The Lancet 397*(10273), 462.

Cappé, O. (2011). Online em algorithm for hidden markov models. *Journal of Computational and Graphical Statistics 20*(3), 728–749.

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

Carvalho, C. M., M. S. Johannes, H. F. Lopes, N. G. Polson, et al. (2010). Particle learning and smoothing. *Statistical Science 25*(1), 88–106.

Casella, G. and C. P. Robert (1996). Rao-Blackwellisation of sampling schemes. *Biometrika 83*(1), 81–94.

Chambers, J. M., C. L. Mallows, and B. Stuck (1976). A method for simulating stable random variables. *Journal of the American Statistical Association 71*(354), 340–344.

Chen, R. and J. S. Liu (2000). Mixture Kalman filters. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 62*(3), 493–508.

Chen, T., T. B. Schon, H. Ohlsson, and L. Ljung (2010). Decentralized particle filter with arbitrary state decomposition. *IEEE Transactions on Signal Processing 59*(2), 465–478.

Chen, Y., Q. Liu, and D. Guo (2020). Emerging coronaviruses: genome structure, replication, and pathogenesis. *Journal of Medical Virology 92*(4), 418–423.

Choi, J. Y. and D. M. Smith (2021). Sars-cov-2 variants of concern. *Yonsei medical journal 62*(11), 961.

Chopin, N., P. E. Jacob, and O. Papaspiliopoulos (2013). Smc2: an efficient algorithm for sequential analysis of state space models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 75*(3), 397–426.

Chopin, N. and S. Singh (2015). On particle Gibbs sampling. *Bernoulli 21*(3), 1855–1883.

Dai, C., J. Heng, P. E. Jacob, and N. Whiteley (2022). An invitation to sequential Monte Carlo samplers. *Journal of the American Statistical Association 117*(539), 1587–1600.

Dehury, B., V. Raina, N. Misra, and M. Suar (2020). Effect of mutation on structure, function and dynamics of receptor binding domain of human sars-cov-2 with host cell receptor ace2: a molecular dynamics simulations study. *Journal of Biomolecular Structure and Dynamics*, 1–15.

Del Moral, P. (1997). Nonlinear filtering: Interacting particle resolution. *Comptes Rendus de l'Académie des Sciences-Series I-Mathematics 325*(6), 653–658.

Del Moral, P., A. Doucet, and A. Jasra (2006). Sequential Monte Carlo samplers. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 68*(3), 411–436.

Dempster, A. P., N. M. Laird, and D. B. Rubin (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 39*(1), 1–22.

Di Lena, P., K. Nagata, and P. Baldi (2012). Deep architectures for protein contact map prediction. *Bioinformatics 28*(19), 2449–2457.

Domingos, P. (2015). *The master algorithm: How the quest for the ultimate learning machine will remake our world*. Basic Books.

Doucet, A., N. De Freitas, and N. Gordon (2001). An introduction to sequential monte carlo methods. In *Sequential Monte Carlo Methods in Practice*, pp. 3–14. Springer.

Drovandi, C. C. and A. N. Pettitt (2011). Likelihood-free Bayesian estimation of multivariate quantile distributions. *Computational Statistics & Data Analysis 55*(9), 2541–2556.

Engle, R. F. (1982). Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation. *Econometrica: Journal of the Econometric Society*, 987–1007.

Engle, R. F. and A. J. Patton (2001). What good is a volatility model? *Quantitative Finance 1*, 237–245.

Esposito, L., A. De Simone, A. Zagari, and L. Vitagliano (2005). Correlation between ω and ψ dihedral angles in protein structures. *Journal of Molecular Biology 347*(3), 483–487.

Fearnhead, P. and P. Clifford (2003). On-line inference for hidden markov models via particle filters. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 65*(4), 887–899.

Francq, C., O. Wintenberger, and J.-M. Zakoian (2013). Garch models without positivity constraints: Exponential or log garch? *Journal of Econometrics 177*(1), 34–46.

Fraser, J. S., M. W. Clarkson, S. C. Degnan, R. Erion, D. Kern, and T. Alber (2009). Hidden alternative structures of proline isomerase essential for catalysis. *Nature 462*(7273), 669–673.

Fraser, J. S., H. van den Bedem, A. J. Samelson, P. T. Lang, J. M. Holton, N. Echols, and T. Alber (2011). Accessing protein conformational ensembles using room-temperature X-ray crystallography. *Proceedings of the National Academy of Sciences 108*(39), 16247–16252.

Fridman, M. and L. Harris (1998). A maximum likelihood approach for non-gaussian stochastic volatility models. *Journal of Business & Economic Statistics 16*(3), 284–291.

Geweke, J. (1986). Modeling the persistence of conditional variances: a comment. *Econometric Reviews 5*, 57–61.

Google LLC (2022). *Google COVID-19 Community Mobility Reports*. Ph. D. thesis, https://www.google.com/covid19/mobility/ Accessed: ¡date¿.

Gordon, N. J., D. J. Salmond, and A. F. Smith (1993). Novel approach to nonlinear/non-gaussian bayesian state estimation. In *IEE Proceedings F (Radar and Signal Processing)*, Volume 140, pp. 107–113. IET.

Harvey, W. T., A. M. Carabelli, B. Jackson, R. K. Gupta, E. C. Thomson, E. M. Harrison, C. Ludden, R. Reeve, A. Rambaut, S. J. Peacock, et al. (2021). Sars-cov-2 variants, spike mutations and immune escape. *Nature reviews microbiology 19*(7), 409–424.

Hastings, W. K. (1970). Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 97–109.

He, Y. and Y.-C. Dong (2022). A perspective on re-detectable positive sars-cov-2 nucleic acid results in recovered covid-19 patients. *Disaster Medicine and Public Health Preparedness 16*(2), 482–486.

Hou, Z. and S. W. Wong (2024). Estimating Boltzmann averages for protein structural quantities using sequential Monte Carlo. *Statistica Sinica 34*, 1263–1280.

Hue, C., J.-P. Le Cadre, and P. Pérez (2002). Sequential monte carlo methods for multiple target tracking and data fusion. *IEEE Transactions on Signal Processing 50*(2), 309–325.

Ioannidis, J. P., S. Cripps, and M. A. Tanner (2022). Forecasting for COVID-19 has failed. *International Journal of Forecasting 38*(2), 423–438.

Jacquier, E., N. G. Polson, and P. E. Rossi (1994). Bayesian analysis of stochastic volatility models. *Journal of Business & Economic Statistics 12*(4), 371–389.

Jacquier, E., N. G. Polson, and P. E. Rossi (2002). Bayesian analysis of stochastic volatility models. *Journal of Business & Economic Statistics 20*(1), 69–87.

Jacquier, E., N. G. Polson, and P. E. Rossi (2004). Bayesian analysis of stochastic volatility models with fat-tails and correlated errors. *Journal of Econometrics 122*(1), 185–212.

Johansen, A. M., N. Whiteley, and A. Doucet (2012). Exact approximation of Rao-Blackwellised particle filters. *IFAC Proceedings Volumes 45*(16), 488–493.

Jun, S.-H., S. W. Wong, J. V. Zidek, A. Bouchard-Côté, et al. (2019). Sequential decision model for inference and prediction on nonuniform hypergraphs with application to knot matching from computational forestry. *Annals of Applied Statistics 13*(3), 1678–1707.

Kabašinskas, A., S. Rachev, L. Sakalauskas, W. Sun, and I. Belovas (2009). Alpha-stable paradigm in financial markets. *Journal of Computational Analysis and Applications 11*(4), 641–668.

Kantas, N., A. Doucet, S. S. Singh, J. Maciejowski, and N. Chopin (2015). On particle methods for parameter estimation in state-space models. *Statistical Science 30*(3), 328–351.

Kanter, M. (1975). Stable densities under change of scale and total variation inequalities. *The Annals of Probability 3*(4), 697–707.

Karlin, S., Z.-Y. Zhu, and F. Baud (1999). Atom density in protein structures. *Proceedings of the National Academy of Sciences 96*(22), 12500–12505.

Keller, J. P., T. Zhou, A. Kaplan, G. B. Anderson, and W. Zhou (2022). Tracking the transmission dynamics of COVID-19 with a time-varying coefficient state-space model. *Statistics in Medicine 41*(15), 2745–2767.

Kerama, I., T. Thorne, and R. G. Everitt (2022). Rare event ABC-SMC^2. *arXiv preprint arXiv:2211.02172*.

Kim, J., S. Jo, and K. Lee (2023). Bayesian computational methods for state-space models with application to SIR model. *Journal of Statistical Computation and Simulation 93*(7), 1207–1223.

Kitagawa, G. (1996). Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *Journal of Computational and Graphical Statistics 5*(1), 1–25.

Kitagawa, G. (1998). A self-organizing state-space mode. *Journal of the American Statistical Association 93*(443), 1203–1215.

Lan, J., J. Ge, J. Yu, S. Shan, H. Zhou, S. Fan, Q. Zhang, X. Shi, Q. Wang, L. Zhang, et al. (2020). Structure of the sars-cov-2 spike receptor-binding domain bound to the ace2 receptor. *Nature 581*(7807), 215–220.

Landau, L. D. and E. M. Lifshitz (2013). *Statistical Physics*, Volume 5. Elsevier.

Lekone, P. E. and B. F. Finkenstädt (2006). Statistical inference in a stochastic epidemic seir model with control intervention: Ebola as a case study. *Biometrics 62*(4), 1170–1177.

Li, Y., W. Wang, K. Deng, and J. S. Liu (2022). Stratification and optimal resampling for sequential Monte Carlo. *Biometrika 109*(1), 181–194.

Ligeti, B., R. Vera, J. Juhász, and S. Pongor (2017). Cx, dpx, and pcw: web servers for the visualization of interior and protruding regions of protein structures in 3d and 1d. In *Prediction of Protein Secondary Structure*, pp. 301–309. Springer.

Lin, M., R. Chen, and J. S. Liu (2013). Lookahead strategies for sequential Monte Carlo. *Statistical Science 28*(1), 69–94.

Lindsten, F., M. I. Jordan, and T. B. Schon (2014). Particle Gibbs with ancestor sampling. *Journal of Machine Learning Research 15*(1), 2145–2184.

Liu, J. and M. West (2001). Combined parameter and state estimation in simulation-based filtering. In *Sequential Monte Carlo Methods in Practice*, pp. 197–223. Springer.

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

Liu, J. S. and R. Chen (1995). Blind deconvolution via sequential imputations. *Journal of the American Statistical Association 90*(430), 567–576.

Liu, J. S. and R. Chen (1998). Sequential monte carlo methods for dynamic systems. *Journal of the American Statistical Association 93*(443), 1032–1044.

Liu, J. S., R. Chen, and T. Logvinenko (2001). A theoretical framework for sequential importance sampling with resampling. In A. Doucet, N. de Freitas, and N. Gordon (Eds.), *Sequential Monte Carlo Methods in Practice*, pp. 225–246. Springer.

Liu, J. S., R. Chen, and W. H. Wong (1998). Rejection control and sequential importance sampling. *Journal of the American Statistical Association 93*(443), 1022–1031.

Liu, L., S. Vikram, J. Lao, X. Ben, A. D'Amour, S. O'Banion, M. Sandler, R. A. Saurous, and M. D. Hoffman (2020). Estimating the changing infection rate of covid-19 using bayesian models of mobility. *medRxiv*, 2020–08.

Lombardi, M. J. and G. Calzolari (2009). Indirect estimation of α-stable stochastic volatility models. *Computational Statistics & Data Analysis 53*(6), 2298–2308.

Lowe, T. E., A. Golightly, and C. Sherlock (2023). Accelerating inference for stochastic kinetic models. *Computational Statistics & Data Analysis 185*, 107760.

Mandelbrot, B. (1963). The variation of certain speculative prices. *The Journal of Business 36*(4), 394–419.

Marin, J.-M., P. Pudlo, C. P. Robert, and R. J. Ryder (2012). Approximate Bayesian computational methods. *Statistics and Computing 22*(6), 1167–1180.

Marks, C., J. Nowak, S. Klostermann, G. Georges, J. Dunbar, J. Shi, S. Kelm, and C. M. Deane (2017). Sphinx: merging knowledge-based and ab initio approaches to improve protein loop prediction. *Bioinformatics 33*(9), 1346–1353.

Metropolis, N., A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller (1953). Equation of state calculations by fast computing machines. *The Journal of Chemical Physics 21*(6), 1087–1092.

Mihel, J., M. Šikić, S. Tomić, B. Jeren, and K. Vlahoviček (2008). Psaia–protein structure and interaction analyzer. *BMC structural biology 8*(1), 1–11.

Milhøj, A. (1987). *A Multiplicative Parameterization of ARCH Models*.

Mittnik, S., T. Doganoglu, and D. Chenyao (1999). Computing the probability density function of the stable paretian distribution. *Mathematical and Computer Modelling 29*(10-12), 235–240.

Moghaddasi, S. S. and N. Faraji (2020). A hybrid algorithm based on particle filter and genetic algorithm for target tracking. *Expert Systems with Applications 147*, 113188.

Murphy, T. J., H. Swail, J. Jain, M. Anderson, P. Awadalla, L. Behl, P. E. Brown, C. L. Charlton, K. Colwill, S. J. Drews, et al. (2023). The evolution of SARS-CoV-2 seroprevalence in Canada: a time-series study, 2020–2023. *CMAJ 195*(31), E1030–E1037.

Neal, R. M. (2001). Annealed importance sampling. *Statistics and Computing 11*, 125–139.

Nguyen, H. L., P. D. Lan, N. Q. Thai, D. A. Nissley, E. P. O'Brien, and M. S. Li (2020). Does sars-cov-2 bind to human ace2 more strongly than does sars-cov? *The Journal of Physical Chemistry B 124*(34), 7336–7347.

Nolan, J. P. (1997). Numerical calculation of stable densities and distribution functions. *Communications in Statistics. Stochastic Models 13*(4), 759–774.

Nolan, J. P. (1999). An algorithm for evaluating stable densities in Zolotarev's (M) parameterization. *Mathematical and Computer Modelling 29*(10-12), 229–233.

Nolan, J. P. (2020). *Univariate Stable Distributions*. Springer.

Onuchic, J. N., Z. Luthey-Schulten, and P. G. Wolynes (1997). Theory of protein folding: the energy landscape perspective. *Annual Review of Physical Chemistry 48*(1), 545–600.

Owen, J., D. J. Wilkinson, and C. S. Gillespie (2015). Scalable inference for Markov processes with intractable likelihoods. *Statistics and Computing 25*, 145–156.

Paige, B. and F. Wood (2016). Inference networks for sequential monte carlo in graphical models. In *International Conference on Machine Learning*, pp. 3040–3049. PMLR.

Pantula, S. G. (1986). Modeling the persistence of conditional variances: a comment. *Econometric Reviews 5*, 79–97.

Peters, G. W., Y. Fan, and S. A. Sisson (2012). On sequential Monte Carlo, partial rejection control and approximate Bayesian computation. *Statistics and Computing 22*, 1209–1222.

Petropoulos, A., S. P. Chatzis, and S. Xanthopoulos (2016). A novel corporate credit rating system based on student'st hidden markov models. *Expert Systems with Applications 53*, 87–105.

Ping, X., Q. Chen, G. Liu, J. Su, and F. Ma (2018). Particle filter based time series prediction of daily sales of an online retailer. In *2018 11th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, pp. 1–6. IEEE.

Pintar, A., O. Carugo, and S. Pongor (2002). CX, an algorithm that identifies protruding atoms in proteins. *Bioinformatics 18*(7), 980–984.

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

Raggi, D. and S. Bordignon (2008). Sequential monte carlo methods for stochastic volatility models with jumps. *Preprint. URL: http://homes. stat. unipd. it/raggi*.

Rayner, G. D. and H. L. MacGillivray (2002). Numerical maximum likelihood estimation for the g-and-k and generalized g-and-h distributions. *Statistics and Computing 12*(1), 57–75.

Rios, M. P. and H. F. Lopes (2013). The extended liu and west filter: Parameter learning in markov switching stochastic volatility models. In *State-Space Models*, pp. 23–61. Springer.

Roberto Telles, C., H. Lopes, and D. Franco (2021). Sars-cov-2: Sir model limitations and predictive constraints. *Symmetry 13*(4), 676.

Roy, A., A. Kucukural, and Y. Zhang (2010). I-tasser: a unified platform for automated protein structure and function prediction. *Nature protocols 5*(4), 725–738.

Sandmann, G. and S. J. Koopman (1998). Estimation of stochastic volatility models via monte carlo maximum likelihood. *Journal of Econometrics 87*(2), 271–301.

Sartorius, B., A. Lawson, and R. Pullan (2021). Modelling and predicting the spatio-temporal spread of covid-19, associated deaths and impact of key risk factors in england. *Scientific reports 11*(1), 5378.

Simons, K. T., C. Kooperberg, E. Huang, and D. Baker (1997). Assembly of protein tertiary structures from fragments with similar local sequences using simulated annealing and bayesian scoring functions. *Journal of Molecular Biology 268*(1), 209–225.

Sipos, I. R., A. Ceffer, and J. Levendovszky (2017). Parallel optimization of sparse portfolios with ar-hmms. *Computational Economics 49*(4), 563–578.

Smith, R. A., E. L. Ionides, and A. A. King (2017). Infectious disease dynamics inferred from genetic data via sequential monte carlo. *Molecular Biology and Evolution 34*(8), 2065–2084.

Stein, A. and T. Kortemme (2013). Improvements to robotics-inspired conformational sampling in rosetta. *PloS one 8*(5), e63090.

Stigler, J., F. Ziegler, A. Gieseke, J. C. M. Gebhardt, and M. Rief (2011). The complex folding network of single calmodulin molecules. *Science 334*(6055), 512–516.

Storvik, G. (2002). Particle filters for state-space models with the presence of unknown static parameters. *IEEE Transactions on Signal Processing 50*(2), 281–289.

Sucarrat, G. (2019). The log-garch model via arma representations. In *Financial Mathematics, Volatility and Covariance Modelling*, pp. 336–359. Routledge.

Svensson, A., T. B. Schön, and M. Kok (2015). Nonlinear state space smoothing using the conditional particle filter. *IFAC-PapersOnLine 48*(28), 975–980.

Tang, K., J. Zhang, and J. Liang (2014). Fast protein loop sampling and structure prediction using distance-guided sequential chain-growth monte carlo method. *PLoS Computational Biology 10*, e1003539.

Taylor, S. J. (2008). *Modelling Financial Time Series*. World scientific.

Tian, T., J. Tan, W. Luo, Y. Jiang, M. Chen, S. Yang, C. Wen, W. Pan, and X. Wang (2021). The effects of stringent and mild interventions for coronavirus pandemic. *Journal of the American Statistical Association 116*(534), 481–491.

Urteaga, I., M. F. Bugallo, and P. M. Djurić (2017). Sequential monte carlo for inference of latent arma time-series with innovations correlated in time. *EURASIP Journal on Advances in Signal Processing 2017*(1), 1–15.

Vankov, E. R., M. Guindani, and K. B. Ensor (2019). Filtering and estimation for a class of stochastic volatility models with intractable likelihoods. *Bayesian Analysis 14*(1), 29–52.

Vermaak, J., M. Gangnet, A. Blake, and P. Perez (2001). Sequential monte carlo fusion of sound and vision for speaker tracking. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, Volume 1, pp. 741–746. IEEE.

Wang, L., S. Wang, and A. Bouchard-Côté (2020). An annealed sequential Monte Carlo method for Bayesian phylogenetics. *Systematic Biology 69*(1), 155–183.

Williams, J. K., B. Wang, A. Sam, C. L. Hoop, D. A. Case, and J. Baum (2021). Molecular dynamics analysis of a flexible loop at the binding interface of the sars-cov-2 spike protein receptor-binding domain. *bioRxiv*.

Wong, K.-C., T.-M. Chan, C. Peng, Y. Li, and Z. Zhang (2013). Dna motif elucidation using belief propagation. *Nucleic Acids Research 41*(16), e153–e153.

Wong, S. W., J. S. Liu, and S. Kou (2017). Fast de novo discovery of low-energy protein loop conformations. *Proteins: Structure, Function, and Bioinformatics 85*(8), 1402–1412.

Wong, S. W., J. S. Liu, and S. Kou (2018). Exploring the conformational space for protein folding with sequential monte carlo. *Annals of Applied Statistics 12*(3), 1628–1654.

Wrapp, D., N. Wang, K. S. Corbett, J. A. Goldsmith, C.-L. Hsieh, O. Abiona, B. S. Graham, and J. S. McLellan (2020). Cryo-EM structure of the 2019-nCoV spike in the prefusion conformation. *Science 367*(6483), 1260–1263.

Wuertz, D., M. Maechler, and Rmetrics core team members. (2016). *stabledist: Stable Distribution Functions*. R package version 0.7-1.

Xie, Y. (2017). *A flexible inference method for an autoregressive stochastic volatility model with an application to risk management*. Ph. D. thesis, University of British Columbia.

Xuan, G., W. Zhang, and P. Chai (2001). Em algorithms of gaussian mixture model and hidden markov model. In *Proceedings 2001 International Conference on Image Processing (Cat. No. 01CH37205)*, Volume 1, pp. 145–148. IEEE.

Yang, B., J. R. Stroud, and G. Huerta (2018). Sequential monte carlo smoothing with parameter estimation. *Bayesian Analysis 13*(4), 1137–1161.

Zarwi, F. E., A. Vij, and J. Walker (2017). Modeling and forecasting the evolution of preferences over time: A hidden markov model of travel behavior. *arXiv preprint arXiv:1707.09133*.

Zhang, J., M. Lin, R. Chen, J. Liang, and J. S. Liu (2007). Monte carlo sampling of near-native structures of proteins with applications. *PROTEINS: Structure, Function, and Bioinformatics 66*(1), 61–68.

Zhao, Y. and S. W. Wong (2023). A comparative study of compartmental models for COVID-19 transmission in Ontario, Canada. *Scientific Reports 13*(1), 15050.

Zheng, W., Y. Li, C. Zhang, R. Pearce, S. Mortuza, and Y. Zhang (2019). Deep-learning contact-map guided protein structure prediction in casp13. *Proteins: Structure, Function, and Bioinformatics 87*(12), 1149–1164.

Zhou, H. and Y. Zhou (2002). Distance-scaled, finite ideal-gas reference state improves structure-derived potentials of mean force for structure selection and stability prediction. *Protein Science 11*(11), 2714–2726.

Zhou, R. and B. Berne (1997). Smart walking: A new method for Boltzmann sampling of protein conformations. *The Journal of Chemical Physics 107*(21), 9185–9196.

APPENDICES

Appendix A

Supplementary Material for Chapter 2

A.1 Pseudocode for SISR algorithm

Require: particle size N;

Initialization: Sample $\{\mathbf{x}_0^{(n)}\}_{n=1}^N$ from $\eta(\mathbf{x}_0)$, and the weight $w(\mathbf{x}_0^{(n)}) = p_0(\mathbf{x}_0^{(n)})/\eta(\mathbf{x}_0^{(n)})$;

for $t = 1, \ldots, T$ **do**

Step 1: Sample $\widetilde{\mathbf{x}}_t$ from $\eta(\mathbf{x}_t \mid \mathbf{x}_{0:t-1}^{(n)})$ and set $\widetilde{\mathbf{x}}_{0:t}^{(n)} = (\mathbf{x}_{0:t-1}^{(n)}, \widetilde{\mathbf{x}}_t)$ for each n;

Step 2: Evaluate the weight $w(\widetilde{\mathbf{x}}_{0:t}^{(n)}) = w(\mathbf{x}_{0:t-1}^{(n)}) p_t(\widetilde{\mathbf{x}}_{0:t}^{(n)})/(p_{t-1}(\mathbf{x}_{0:t-1}^{(n)})\eta(\widetilde{\mathbf{x}}_t \mid \mathbf{x}_{0:t-1}^{(n)}))$ for each n;

Step 3: Resample N particles $\{\mathbf{x}_{0:t}^{(n)}\}_{n=1}^N$ from $\{\widetilde{\mathbf{x}}_{0:t}^{(n)}\}_{n=1}^N$ based on $\{w(\widetilde{\mathbf{x}}_{0:t}^{(n)})\}_{n=1}^N$ and update the weights $\{w(\mathbf{x}_{0:t}^{(n)})\}_{n=1}^N$;

Sequential Importance Sampling with Resampling (SISR) is a common framework to implement propagation with the help of importance distributions $\eta(\mathbf{x}_0), \eta(\mathbf{x}_1 \mid \mathbf{x}_0), \ldots, \eta(\mathbf{x}_T \mid \mathbf{x}_{0:T-1})$ and resampling (Liu and Chen, 1995, 1998). To briefly note some key features of SISR in the algorithm above:

1. **Step 1**: In this step, one descendant is sampled for each particle.
2. **Step 2**: In this step, propagation occurs with the help of importance distributions. The importance weights are usually uneven in practice.

3. **Step 3**: This step involves resampling from the propagated particles. If Step 3 is omitted, the SISR framework reduces to sequential importance sampling (SIS).

The necessity of Step 3 depends on the importance weights: if the importance weights are all constant, resampling only reduces the distinction of the particles and thus increases Monte Carlo variance. However, the importance weights are usually uneven in practice; then without Step 3, some of the importance weights evolving in Step 2 may eventually decay to zero during propagation, which is known as weight degeneracy.

A.2 Proper weighting of the proposed sampling scheme

Recall $p_t(\mathbf{x}_{0:t})$ denotes the auxiliary distribution for step t in Section 3.2 in this context. Let $\mathcal{S}_0 = \{\mathbf{x}_0^{(n_0)}; n_0 = 1, \ldots, NM\}$ and $\mathcal{S}_t = \mathcal{S}_{t-1} \cup \{(\mathbf{x}_0^{(n_0)}, \ldots, \mathbf{x}_t^{(n_t)}); n_0 = 1, \ldots, NM; n_1 = 1, \ldots, M; \ldots; n_t = 1, \ldots, M\}$, i.e., $\mathcal{S}_t$ is the collection of upsampled particles up to time t. Any $\mathbf{x}_{0:t}^* = (\mathbf{x}_0^{(n_0^*)}, \ldots, \mathbf{x}_t^{(n_t^*)}) \in \mathcal{S}_t$ satisfies $\mathbf{x}_{0:s}^* = (\mathbf{x}_0^{(n_0^*)}, \ldots, \mathbf{x}_s^{(n_s^*)}) \in \mathcal{S}_t$ for $s \leq t$. Let $\eta(\mathbf{x}_{0:t})$ denote $\eta(\mathbf{x}_0) \prod_{s=1}^t \eta(\mathbf{x}_s \mid \eta(\mathbf{x}_{0:s-1})$ and $\mathcal{Q}_{0:t}$ denote the σ-algebra generated by $Q(\mathbf{x}_{0:t})$ conditional on $\mathcal{S}_t$.

When $t = 0$, it is obvious that $E_{\mathcal{Q}_0}\left[Q(\mathbf{x}_0^*) \mid \mathcal{S}_0\right] = \frac{p_0(\mathbf{x}_0^*)}{\eta(\mathbf{x}_0^*)}$ and

$$E_\eta\left\{E_{\mathcal{Q}_0}\left[Q(\mathbf{x}_0^*) \mid \mathcal{S}_0\right]\right\} = E_\eta\left\{\frac{p_0(\mathbf{x}_0^*)}{\eta(\mathbf{x}_0^*)}\right\} = \int \frac{p_0(\mathbf{x}_0^*)}{\eta(\mathbf{x}_0^*)} \eta(\mathbf{x}_0^*) d\mathbf{x}_{0:t}^* = 1$$

and for any square integrable function $f_0(\mathbf{x}_0)$,

$$\begin{aligned} E_\eta\left\{E_{\mathcal{Q}_0}\left[f_0(\mathbf{x}_0^*)Q(\mathbf{x}_0^*) \mid \mathcal{S}_0\right]\right\} &= E_\eta\left\{f_0(\mathbf{x}_0^*)E_{\mathcal{Q}_0}\left[Q(\mathbf{x}_0^*) \mid \mathcal{S}_0\right]\right\} \\ &= E_\eta\left\{f_0(\mathbf{x}_0^*)\frac{p_0(\mathbf{x}_0^*)}{\eta(\mathbf{x}_0^*)}\right\} \\ &= \int f_0(\mathbf{x}_0^*)\frac{p_0(\mathbf{x}_0^*)}{\eta(\mathbf{x}_0^*)}\eta(\mathbf{x}_0^*)d\mathbf{x}_0^* \\ &= E_{p_0}\left[h(\mathbf{x}_0^*)\right], \end{aligned}$$

which justifies the proper weighting condition.

We now proceed by induction: assume $E_{\mathcal{Q}_{0:t-1}}\left[Q(\mathbf{x}_{0:t-1}^*) \mid \mathcal{S}_{t-1}\right] = \frac{p_{t-1}(\mathbf{x}_{0:t-1}^*)}{\eta(\mathbf{x}_{0:t-1}^*)}$ holds. Conditional on $\mathcal{S}_t$ and $Q(\mathbf{x}_{0:t-1})$ for all $\mathbf{x}_{0:t-1} \in \mathcal{S}_{t-1}$, and our SMC produces $Q(\mathbf{x}_{0:t}^*)$ as

$$Q(\mathbf{x}_{0:t}^*) = \begin{cases} \frac{Q(\mathbf{x}_{0:t-1}^*)p_t(\mathbf{x}_{0:t}^*)}{p_{t-1}(\mathbf{x}_{0:t-1}^*)\eta(\mathbf{x}_t^* \mid \mathbf{x}_{0:t-1}^*)} \frac{1}{q(\mathbf{x}_{0:t}^*)} & \text{with probability } q(\mathbf{x}_{0:t}^*) \\ 0 & \text{otherwise} \end{cases}$$

where $q(\mathbf{x}^*_{0:t}) = \min\{\frac{c_t Q(\mathbf{x}^*_{0:t-1}) p_t(\mathbf{x}^*_{0:t})}{p_{t-1}(\mathbf{x}^*_{0:t-1})\eta(\mathbf{x}^*_t|\mathbf{x}^*_{0:t-1})}, 1\}$ with c_t being the root of

$$\sum_{\mathbf{x}_{0:t}\in\mathcal{S}_t} \min\left\{\frac{c_t Q(\mathbf{x}_{0:t-1}) p_t(\mathbf{x}_{0:t})}{p_{t-1}(\mathbf{x}_{0:t-1})\eta(\mathbf{x}_t \mid \mathbf{x}_{0:t-1})}, 1\right\} = N.$$

We can rewrite $Q(\mathbf{x}^*_{0:t})$ as

$$Q(\mathbf{x}^*_{0:t}) = \frac{Q(\mathbf{x}^*_{0:t-1}) p_t(\mathbf{x}^*_{0:t})}{p_{t-1}(\mathbf{x}^*_{0:t-1})\eta(\mathbf{x}^*_t \mid \mathbf{x}^*_{0:t-1})} \frac{1}{q(\mathbf{x}^*_{0:t})} \mathbf{I}(\mathbf{x}^*_{0:t}),$$

where $\mathbf{I}(\mathbf{x}^*_{0:t})$ denotes an indicator function defined conditional on $\mathcal{S}_t$ and $Q(\mathbf{x}^*_{0:t-1})$ with $E_{\mathbf{I}}[\mathbf{I}(\mathbf{x}^*_{0:t}) \mid Q(\mathbf{x}^*_{0:t-1}), \mathcal{S}_t] = q(\mathbf{x}^*_{0:t})$. Note that $\mathbf{I}(\mathbf{x}^*_{0:t})$ is independent of future descendants, so for $0 < r < s \leq t$,

$$E_{\mathbf{I}}\left[\mathbf{I}(\mathbf{x}^*_{0:r}) \mid Q(\mathbf{x}^*_{0:r-1}), \mathcal{S}_s\right] = E_{\mathbf{I}}\left[\mathbf{I}(\mathbf{x}^*_{0:r}) \mid Q(\mathbf{x}^*_{0:r-1}), \mathcal{S}_r\right].$$

Now we can see that $Q(\mathbf{x}^*_{0:t})$ is proportional to the product of the two random variables $Q(\mathbf{x}^*_{0:t-1})$ and $\mathbf{I}(\mathbf{x}^*_{0:t})$, with $\mathbf{I}(\mathbf{x}^*_{0:t})$ conditional on $Q(\mathbf{x}^*_{0:t-1})$. By the tower rule, we have

$$\begin{aligned}
E_{\mathcal{Q}_{0:t}}\left\{Q(\mathbf{x}^*_{0:t}) \mid \mathcal{S}_t\right\} &= E_{\mathcal{Q}_{0:t-1}}\left\{\frac{Q(\mathbf{x}^*_{0:t-1}) p_t(\mathbf{x}^*_{0:t})}{p_{t-1}(\mathbf{x}^*_{0:t-1})\eta(\mathbf{x}^*_t \mid \mathbf{x}^*_{0:t-1})} \frac{1}{q(\mathbf{x}^*_{0:t})} E_{\mathbf{I}}\left[\mathbf{I}(\mathbf{x}^*_{0:t}) \mid Q(\mathbf{x}^*_{0:t-1}), \mathcal{S}_t\right] \mid \mathcal{S}_t\right\} \\
&= E_{\mathcal{Q}_{0:t-1}}\left[\frac{Q(\mathbf{x}^*_{0:t-1}) p_t(\mathbf{x}^*_{0:t})}{p_{t-1}(\mathbf{x}^*_{0:t-1})\eta(\mathbf{x}^*_t \mid \mathbf{x}^*_{0:t-1})} \mid \mathcal{S}_t\right] \\
&= E_{\mathcal{Q}_{0:t-1}}\left[Q(\mathbf{x}^*_{0:t-1}) \mid \mathcal{S}_t\right] \frac{p_t(\mathbf{x}^*_{0:t})}{p_{t-1}(\mathbf{x}^*_{0:t-1})\eta(\mathbf{x}^*_t \mid \mathbf{x}^*_{0:t-1})}.
\end{aligned}$$

Note that $Q(\mathbf{x}^*_{0:t-1})$ is only conditional on $\mathcal{S}_{t-1}$ and thus independent of future descendants, so

$$E_{\mathcal{Q}_{0:t-1}}\left[Q(\mathbf{x}^*_{0:t-1}) \mid \mathcal{S}_t\right] = E_{\mathcal{Q}_{0:t-1}}\left[Q(\mathbf{x}^*_{0:t-1}) \mid \mathcal{S}_{t-1}\right]$$

and thus

$$E_{\mathcal{Q}_{0:t}}\left\{Q(\mathbf{x}^*_{0:t}) \mid \mathcal{S}_t\right\} = E_{\mathcal{Q}_{0:t-1}}\left[Q(\mathbf{x}^*_{0:t-1}) \mid \mathcal{S}_{t-1}\right] \frac{p_t(\mathbf{x}^*_{0:t})}{p_{t-1}(\mathbf{x}^*_{0:t-1})\eta(\mathbf{x}^*_t \mid \mathbf{x}^*_{0:t-1})} = \frac{p_t(\mathbf{x}^*_{0:t})}{\eta(\mathbf{x}^*_{0:t})}.$$

Therefore,

$$E_\eta\left\{E_{\mathcal{Q}_{0:t}}\left[Q(\mathbf{x}^*_{0:t}) \mid \mathcal{S}_t\right]\right\} = E_\eta\left\{\frac{p_t(\mathbf{x}^*_{0:t})}{\eta(\mathbf{x}^*_{0:t})}\right\} = \int \frac{p_t(\mathbf{x}^*_{0:t})}{\eta(\mathbf{x}^*_{0:t})} \eta(\mathbf{x}^*_{0:t}) d\mathbf{x}^*_{0:t} = 1$$

and for any square integrable function $f_t(\mathbf{x}_{0:t})$,

$$\begin{aligned} E_\eta \left\{ E_{\mathcal{Q}_{0:t}} \left[f_t(\mathbf{x}^*_{0:t}) Q(\mathbf{x}^*_{0:t}) \mid \mathcal{S}_t \right] \right\} &= E_\eta \left\{ f_t(\mathbf{x}^*_{0:t}) E_{\mathcal{Q}_{0:t}} \left[Q(\mathbf{x}^*_{0:t}) \mid \mathcal{S}_t \right] \right\} \\ &= E_\eta \left\{ f_t(\mathbf{x}^*_{0:t}) \frac{p_t(\mathbf{x}^*_{0:t})}{\eta(\mathbf{x}^*_{0:t})} \right\} \\ &= \int f_t(\mathbf{x}^*_{0:t}) \frac{p_t(\mathbf{x}^*_{0:t})}{\eta(\mathbf{x}^*_{0:t})} \eta(\mathbf{x}^*_{0:t}) d\mathbf{x}^*_{0:t} \\ &= E_{p_t} \left[f_t(\mathbf{x}_{0:t}) \right], \end{aligned}$$

which justifies the proper weighting condition.

A.3 Minimization of the conditional expected squared error loss

Fearnhead and Clifford (2003) have shown that in the downsampling step, only N of the $Q(\mathbf{x}^{(n,m)}_{0:t})$'s are non-zero so that for some random variable $C^{(n,m)}_t$ with σ-algebra $\mathcal{C}$, we have

$$Q(\mathbf{x}^{(n,m)}_{0:t}) = \begin{cases} C^{(n,m)}_t & \text{with probability } q(\mathbf{x}^{(n,m)}_{0:t}) \\ \\ 0 & \text{otherwise} \end{cases}$$

Assume the set of particles $\{\mathbf{x}^{(n,m)}_{0:t}, n = 1, \ldots, N \text{ and } m = 1, \ldots, M\}$ are obtained after upsampling and let $\gamma^{(n,m)}_t = \left(p_t(\mathbf{x}^{(n,m)}_{0:t}) / \eta(\mathbf{x}^{(n,m)}_{0:t}) \right) / \sum_{n=1}^N \sum_{m=1}^M \left(p_t(\mathbf{x}^{(n,m)}_{0:t}) / \eta(\mathbf{x}^{(n,m)}_{0:t}) \right)$, then the expected squared error loss for $Q(\mathbf{x}^{(n,m)}_{0:t})$ can be written as

$$\begin{aligned} E_{\mathcal{Q}_{0:t}} \left[(Q(\mathbf{x}^{(n,m)}_{0:t}) - \gamma^{(n,m)}_t)^2 \mid \mathcal{S}_t \right] &= E_{\mathcal{C}} \left\{ E_{\mathcal{Q}_{0:t}} \left[(Q(\mathbf{x}^{(n,m)}_{0:t}) - \gamma^{(n,m)}_t)^2 \mid \mathcal{S}_t, C^{(n,m)} \right] \middle| \mathcal{S}_t \right\} \\ &= q(\mathbf{x}^{(n,m)}_{0:t}) \{ E_{\mathcal{C}} (C^{(n,m)} - \gamma^{(n,m)}_t \mid \mathcal{S}_t) \}^2 \\ &\quad + q(\mathbf{x}^{(n,m)}_{0:t}) \mathrm{Var}(C^{(n,m)} \mid \mathcal{S}_t) + (1 - q(\mathbf{x}^{(n,m)}_{0:t})) {\gamma^{(n,m)}_t}^2, \end{aligned}$$

which is minimized when $C^{(n,m)}$ is a constant for each n and m. To ensure proper weighting as shown in Section S2 of the supplement, we set $C^{(n,m)} = w(\mathbf{x}^{(n,m)}_{0:t}) / q(\mathbf{x}^{(n,m)}_{0:t})$ where $w(\mathbf{x}^{(n,m)}_{0:t})$ denotes the upsampling weight of $\mathbf{x}^{(n,m)}_{0:t}$, and then the conditional expected squared error loss is minimized subject to $\sum_{n=1}^N \sum_{m=1}^M q(\mathbf{x}^{(n,m)}_{0:t}) \leq N$, since

$$\min \left\{ E_{\mathcal{Q}_{0:t}} \left[\sum_{n=1}^N \sum_{m=1}^M (Q(\mathbf{x}^{(n,m)}_{0:t}) - \gamma^{(n,m)}_t)^2 \middle| \mathcal{S}_t \right] \right\} = \sum_{n=1}^N \sum_{m=1}^M (w(\mathbf{x}^{(n,m)}_{0:t}) / q(\mathbf{x}^{(n,m)}_{0:t}) - \gamma^{(n,m)}_t)^2.$$

A.4 Details of the energy function

Given a target segment represented by the dihedral angles $\mathbf{x}_{0:T}$ and for $t \in \{1, \ldots, T\}$, the details of the two components of the energy function H are as follows:

- *Atomic interactions* – the energy of atomic interactions (denoted by H_a) is derived from the pairwise distances between atoms. Notably, we exclude atom pairs that either share bonds or belong to the same amino acid. To calculate this energy, we employ the distance-scaled, finite ideal-gas reference (DFIRE) state potential, as proposed by Zhou and Zhou (2002). DFIRE assigns a score to the pairwise distance for each atom type. Additionally, we calculate the pairwise distance between the current amino acid $\mathbf{x}_t$ and the C_α atom after the end of the segment (i.e., C_α^{T+2}) to determine whether the segment will satisfy the constraint of forming a continuous backbone with the rest of the protein (Wong et al., 2017). In cases where there are *steric clashes* (i.e., overlapping atoms in close proximity defined by a value of 8 in the DFIRE table), or when the conformation cannot form a continuous backbone with the rest of the protein, we assign $H_a = +\infty$.

  Let $\mathbf{A}_0$ denote the collection of all the atoms in the protein, excluding those in the segment $\mathbf{x}_{0:T}$. For a given partially sampled conformation $\mathbf{x}_{0:t-1}$, the subsequent propagation step requires evaluating the incremental energy of $\mathbf{x}_t$, denoted as $H(\mathbf{x}_t \mid \mathbf{x}_{0:t-1})$. To calculate this, we compute and accumulate the energy scores of the four backbone atoms C^t, O^t, N^{t+1}, and ${C_\alpha}^{t+1}$. For instance, the energy of N^{t+1}, denoted by $En(N^{t+1} \mid \mathbf{x}_{0:t-1})$, is the cumulative energy score between N^{t+1} and (a) all atoms in $\mathbf{A}_0$, (b) all atoms in $\mathbf{x}_{0:t-1}$. The DFIRE potential provides a lookup table for energy scores based on (i) the types of two atoms and (ii) the Euclidean distance between the two atoms, i.e., $\text{DFIRE}(\text{atom}_1, \text{atom}_2)$ can be defined as a function of the atom type of atom_1, the atom type of atom_2, and the Euclidean distance between atom_1 and atom_2. Therefore,

$$\begin{aligned} En(N^{t+1} \mid \mathbf{x}_{0:t-1}) = \sum_{s=0}^{t-1} \big[&\text{DFIRE}(N^{t+1}, C^s) + \text{DFIRE}(N^{t+1}, O^s) + \\ &\text{DFIRE}(N^{t+1}, N^{s+1}) + \text{DFIRE}(N^{t+1}, C_\alpha^{s+1})\big] \\ &+ \sum_{A \in \mathbf{A}_0} \text{DFIRE}(N^{t+1}, A). \end{aligned} \tag{A.1}$$

  Likewise, we can compute $En(C^t \mid \mathbf{x}_{0:t-1})$, $En(O^t \mid \mathbf{x}_{0:t-1})$ and $En(C_\alpha^{t+1} \mid \mathbf{x}_{0:t-1})$ by substituting N^{t+1} with the corresponding atom in (A.1). $H_a(\mathbf{x}_t \mid \mathbf{x}_{0:t-1})$ is then

assigned by

$$H_a(\mathbf{x}_t \mid \mathbf{x}_{0:t-1}) = En(\mathrm{C}^t \mid \mathbf{x}_{0:t-1}) + En(\mathrm{O}^t \mid \mathbf{x}_{0:t-1}) + En(\mathrm{N}^{t+1} \mid \mathbf{x}_{0:t-1}) + En(\mathrm{C}_\alpha^{t+1} \mid \mathbf{x}_{0:t-1}).$$

- *Dihedral angles* – the energy of backbone dihedral angles (denoted by H_θ) is calculated from the empirical distributions $\tilde{p}(\phi_t, \psi_t, \omega_t)$ derived from historical protein data, independently for each amino acid with

  $$\tilde{p}(\phi_t, \psi_t, \omega_t) = \tilde{p}(\phi_t, \psi_t)\tilde{p}(\omega_t),$$

  where $\tilde{p}(\phi_t, \psi_t)$ is based on the amino acid type of a_t and consists of discrete bins of every 5° for each angle (which we assume to be uniformly distributed within each bin), i.e., each empirical distribution is stored in a 72 by 72 matrix; $\tilde{p}(\omega_t)$ is a Gaussian distribution with mean 180° and standard deviation 3° independent of (ϕ_t, ψ_t). Following that work of Wong et al. (2017), we then set $H_\theta(\mathbf{x}_t) = -\log[\tilde{p}(\phi_t, \psi_t, \omega_t)]$.

Then the incremental energy of $\mathbf{x}_t$ is defined as

$$H(\mathbf{x}_t \mid \mathbf{x}_{0:t-1}) = H_a(\mathbf{x}_t \mid \mathbf{x}_{0:t-1}) + H_\theta(\mathbf{x}_t). \tag{A.2}$$

The auxiliary distribution at t is

$$p_t(\mathbf{x}_{0:t}) \propto \exp\left(-H_a(\mathbf{x}_0)\right)\tilde{p}(\mathbf{x}_0) \prod_{s=1}^{t} \exp\left(-H_a(\mathbf{x}_s \mid \mathbf{x}_{0:s-1})\right)\tilde{p}(\mathbf{x}_s).$$

It is convenient to set $\eta(\mathbf{x}_t \mid \mathbf{x}_{0:t-1}) = \tilde{p}(\mathbf{x}_t)$ as the importance distribution: to draw $\mathbf{x}_t$ from $\tilde{p}(\mathbf{x}_t)$, we sample a bin for (ϕ, ψ) from its empirical distribution then draw a (ϕ_t, ψ_t) value uniformly from the selected bin, and ω_t is drawn from its normal distribution. The equation for upsampled weights thus simplifies to

$$\begin{aligned} w(\mathbf{x}_{0:t}) &= w(\mathbf{x}_{0:t-1}) \frac{p_t(\mathbf{x}_{0:t})}{p_{t-1}(\mathbf{x}_{0:t-1})\eta(\mathbf{x}_t \mid \mathbf{x}_{0:t-1})} \\ &= w(\mathbf{x}_{0:t-1}) \exp\left(-H_a(\mathbf{x}_t \mid \mathbf{x}_{0:t-1})\right). \end{aligned} \tag{A.3}$$

A.5 UDSMC experiment for different values of M

The performance of UDSMC will be influenced by the choice of the upsample size M. For a finite space $\mathcal{X}$, a natural choice for M can be to take $M = |\mathcal{X}|$; however, there may not be an intuitive choice for M for a continuous space $\mathcal{X}$. Thus to choose a reasonable value of M for our application, in this section we run a preliminary experiment using different values of M (with a fixed computational budget, i.e., holding MN constant).

For the experiment, we focus on sampling a length 10 segment at amino acid positions 282–291 of the protein with PDB ID *1ds1A*, and thus the dimension of the Boltzmann distribution is 30 in this case. The protein structural quantities of interest are the Boltzmann averages for the number of atomic contacts of C_α at each of the positions 283–292, denoted as $n({C_\alpha}^{283}), \ldots, n({C_\alpha}^{292})$. We vary the values of N and M in the experiment, using $NM \in \{10^5, 5 \times 10^5, 10^6\}$ and $M \in \{1, 5, 10, 20, 50, 100, 200, 500, 1000\}$. For each combination of N and M, we conduct 100 independent repetitions of UDSMC. Notably, when $M = 1$, UDSMC is equivalent to SISR.

Table A.1 presents the Monte Carlo variance of the estimates for different values of M. While RMSEs to the ground truth would be more informative, the ground truth is very expensive to compute, and Monte Carlo variance can still provide useful guidance for a reasonable choice of M. When M is too large or too small (e.g., $M \in \{1, 200, 500, 1000\}$), some of the 100 repetitions end prematurely due to all propagated particles having zero weights (i.e., $H_a = \infty$). A small M can lead to many duplicate particles among the N particles after each propagation step, resulting in insufficient exploration of the target distribution, which may eventually lead to all dead ends after successive iterations of propagation and resampling. On the other hand, a large M implies a small particle size N (for the same computational cost), which may be inadequate to avoid dead ends.

For values of M in the range of $\{5, 10, 20, 50, 100\}$, all UDSMC repetitions completed successfully, and we report the variance of the 100 estimates for each structural quantity. With M fixed, the Monte Carlo variances of the estimates decreased as NM increased. Furthermore, with NM fixed, the variances tended to decrease as M increased from 5 to 20, and then increased again as M increased from 20 to 100. These results highlight the need for a balance between N and M given a fixed computational budget. In this specific experiment, $M = 20$ produced the lowest variances and can be a reasonable choice for the protein sampling problem, and we adopt this value in the simulation study and the real data example.

Table A.1: Variance of the UDSMC estimates over 100 repetitions when estimating the Boltzmann average of the atomic contacts of C_α at each position from 283 to 292, for different combinations of NM and M. "NA" indicates that the algorithm failed to complete for that combination of N and M.

	M	$n(C_\alpha^{283})$	$n(C_\alpha^{284})$	$n(C_\alpha^{285})$	$n(C_\alpha^{286})$	$n(C_\alpha^{287})$	$n(C_\alpha^{288})$	$n(C_\alpha^{289})$	$n(C_\alpha^{290})$	$n(C_\alpha^{291})$	$n(C_\alpha^{292})$
$NM = 100000$	1,2	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
	5	0.277	1.557	3.547	10.004	3.525	3.467	31.481	8.299	4.119	0.599
	10	0.160	0.959	3.378	10.335	2.525	4.194	25.199	10.376	3.843	0.680
	20	0.181	0.814	2.695	7.421	2.633	2.269	12.973	9.321	3.119	0.453
	50	0.313	1.908	4.237	10.802	3.890	3.157	19.476	10.196	4.070	0.576
	100	0.428	3.117	7.379	20.195	6.050	6.695	26.506	14.656	5.471	0.792
	200,500,1000	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
$NM = 500000$	1,2	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
	5	0.083	0.671	1.198	5.014	1.179	1.610	13.895	3.041	1.417	0.346
	10	0.063	0.560	1.183	3.489	1.167	1.421	9.391	3.400	1.443	0.258
	20	0.034	0.202	0.578	2.042	0.696	0.749	3.629	2.828	0.856	0.113
	50	0.061	0.345	1.379	2.542	0.945	0.799	4.270	2.824	1.105	0.136
	100	0.139	0.697	1.812	5.946	1.696	2.192	8.269	5.148	1.355	0.228
	200,500,1000	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
$NM = 1000000$	1,2	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
	5	0.051	0.374	0.637	2.504	0.523	1.058	6.682	1.914	0.590	0.096
	10	0.035	0.234	0.499	1.741	0.505	0.860	5.571	1.258	0.527	0.147
	20	0.024	0.132	0.556	0.762	0.315	0.226	1.222	1.327	0.388	0.070
	50	0.036	0.179	0.535	1.421	0.521	0.482	2.498	1.504	0.512	0.065
	100	0.058	0.369	1.070	2.529	0.744	1.088	3.364	2.319	0.857	0.129
	200,500,1000	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA

A.6 Pseudocode for SISR with one-step lookahead

sample $\{X_0^{(n)}, n = 1, \ldots, N\}$ from the proposal distribution $\eta(X_0)$;
for *n in 1:N* **do**
  sample L descendants of $X_0^{(n)}$ and form $\{X_{0:1}^{(n,l)}, l = 1, \ldots, L\}$ from the proposal distribution $\eta(X_1 \mid X_0^{(n)})$;
  set $\widetilde{w}_{0:1}(X_{0:1}^{(n,l)}) = \frac{p_{0:1}(X_{0:1}^{(n,l)})}{\eta(X_0^{(n,m)})\eta(X_1^{(l)} \mid X_0^{(n)})}$ and $w_{0:1}(X_{0:1}^{(n,l)}) = \widetilde{w}_{0:1}(X_{0:1}^{(n,l)}) / \sum_{l=1}^{L} \widetilde{w}_{0:1}(X_{0:1}^{(n,l)})$
resample particles $\{\tilde{X}_0^{(n)}, n = 1, \ldots, N\}$ proportional to $\{w_{0:1}(X_{0:1}^{(n,l)}), n = 1, \ldots, N, l = 1, \ldots, L\}$;
set $\{X_0^{(n)}, n = 1, \ldots, N\} = \{\tilde{X}_0^{(n)}, n = 1, \ldots, N\}$ with weights $\{W_0^{(n)}, n = 1, \ldots, N\}$;
for *t in 1:T-2* **do**
  for *n in 1:N* **do**
    sample a descendants X_t^* of $X_{t-1}^{(n)}$ and form $X_{0:t}^{(n)} = (X_{t-1}^{(n)}, X_t^*)$ from the proposal distribution $\eta(X_t \mid X_{t-1}^{(n)})$;
    sample L descendants of $X_{0:t}^{(n)}$ and form $\{X_{0:t+1}^{(n,l)}, l = 1, \ldots, L\}$ from the proposal distribution $\eta(X_{t+1} \mid X_{0:t}^{(n)})$;
    set $\widetilde{w}_{0:t+1}(X_{0:t+1}^{(n,l)}) = W_{0:t-1}^{(n)} \frac{p_{0:t+1}(X_{0:t+1}^{(n,l)})}{p_{0:t-1}(X_{0:t-1}^{(n)})\eta(X_t^* \mid X_{0:t-1}^{(n)})\eta(X_{t+1}^{(l)} \mid X_{0:t}^{(n)})}$ and $w_{0:t+1}(X_{0:t+1}^{(n,l)}) = \widetilde{w}_{0:t+1}(X_{0:t+1}^{(n,l)}) / \sum_{l=1}^{L} \widetilde{w}_{0:t+1}(X_{0:t+1}^{(n,l)})$;
  resample particles $\{\tilde{X}_{0:t}^{(n)}, n = 1, \ldots, N\}$ proportional to $\{w_{0:t+1}(X_{0:t+1}^{(n,l)}), n = 1, \ldots, N, l = 1, \ldots, L\}$;
  set $\{X_{0:t}^{(n)}, n = 1, \ldots, N\} = \{\tilde{X}_{0:t}^{(n)}, n = 1, \ldots, N\}$ with weights $\{W_{0:t}^{(n)}, n = 1, \ldots, N\}$;
for *n in 1:N* **do**
  sample a descendants X_{T-1}^* of $X_{0:T-2}^{(n)}$ and form $X_{0:T-1}^{(n)} = (X_{T-2}^{(n)}, X_{0:T-1}^*)$ from the proposal distribution $\eta(X_{T-1} \mid X_{T-2}^{(n)})$;
  sample L descendants of $X_{0:T-1}^{(n)}$ and form $\{X_{0:T}^{(n,l)}, l = 1, \ldots, L\}$ from the proposal distribution $\eta(X_T \mid X_{0:T-1}^{(n)})$;
  set $\widetilde{w}_{0:T}(X_{0:T}^{(n,l)}) = W_{0:T-2}^{(n)} \frac{p_{0:T}(X_{0:T}^{(n,l)})}{p_{0:T-2}(X_{0:T-2}^{(n)})\eta(X_{T-1}^* \mid X_{0:T-2}^{(n)})\eta(X_T^{(l)} \mid X_{0:T-1}^{(n)})}$ and $w_{0:T}(X_{0:T}^{(n,l)}) = \widetilde{w}_{0:T}(X_{0:T}^{(n,l)}) / \sum_{l=1}^{L} \widetilde{w}_{0:T}(X_{0:T}^{(n,l)})$;
resample particles $\{\tilde{X}_{0:T}^{(n)}, n = 1, \ldots, N\}$ proportional to $\{w_{0:T}(X_{0:T}^{(n,l)}), n = 1, \ldots, N, l = 1 \ldots, L\}$;
set $\{X_{0:T}^{(n)}, n = 1, \ldots, N\} = \{\tilde{X}_{0:T}^{(n)}, n = 1, \ldots, N\}$ with weights $\{W_{0:T}^{(n)}, n = 1, \ldots, N\}$;

A.7 Particle diversity of UDSMC and SISR

The resampling step of the t-th SMC iteration ($t = 0, \ldots, 7$) produces N particles for $\mathbf{x}_{0:t}$. Here, we focus on the values of $(\phi_0, \psi_0, \ldots, \phi_t, \psi_t)$ since the ω dihedral angle is usually close to 180°. Thus, for example, we can examine how the values of (ϕ_0, ψ_0) among resampled particles evolve for $t = 0, \ldots, 7$. Similarly, for any $s = 1, \ldots, 7$, we can examine how the values of (ϕ_s, ψ_s) among resampled particles evolve for $t = s, \ldots, 7$. Due to the possibility of weight degeneracy, the diversity of these values (and hence the diversity of the paths

of the particles) may decay as t increases. We compare UDSMC ($N = 20000, M = 20$) and SISR ($N = 400000$) by plotting these values in Figures A.1 and A.2 for one repetition of the simulation study in Section 4 of the main text. The plots show that the diversity of these values decays much faster for SISR. Taking ϕ_0 as an example, both UDSMC and SISR have values that are similarly diverse up to $t = 3$, and subsequently for $t = 4, \ldots, 7$, only UDSMC is able to maintain diversity in ϕ_0 (while SISR decays to just a few unique values). A similar pattern can be seen for other ϕ_s and ψ_s. In this sense, UDSMC is much better at mitigating decay in the diversity of the paths of the particles compared to SISR, which provides an intuitive explanation for the better performance of UDSMC in this application.

Figure A.1: Values of ϕ_s (left panels) and ψ_s (right panels) among resampled particles, for $s = 0, 1, 2, 3$ after each iteration (t) of UDSMC (blue) and SISR (red), for the protein segment in the simulation study.

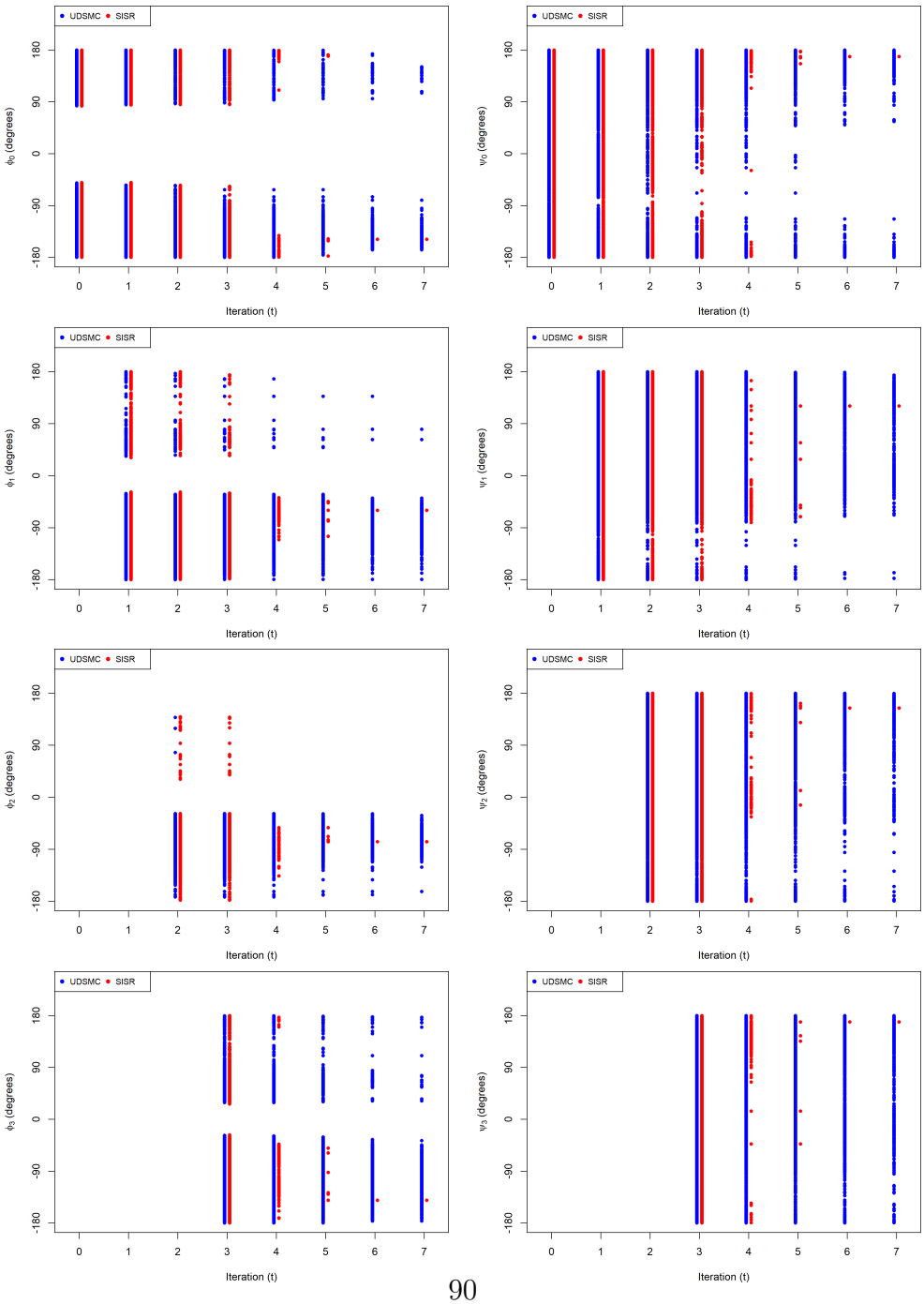
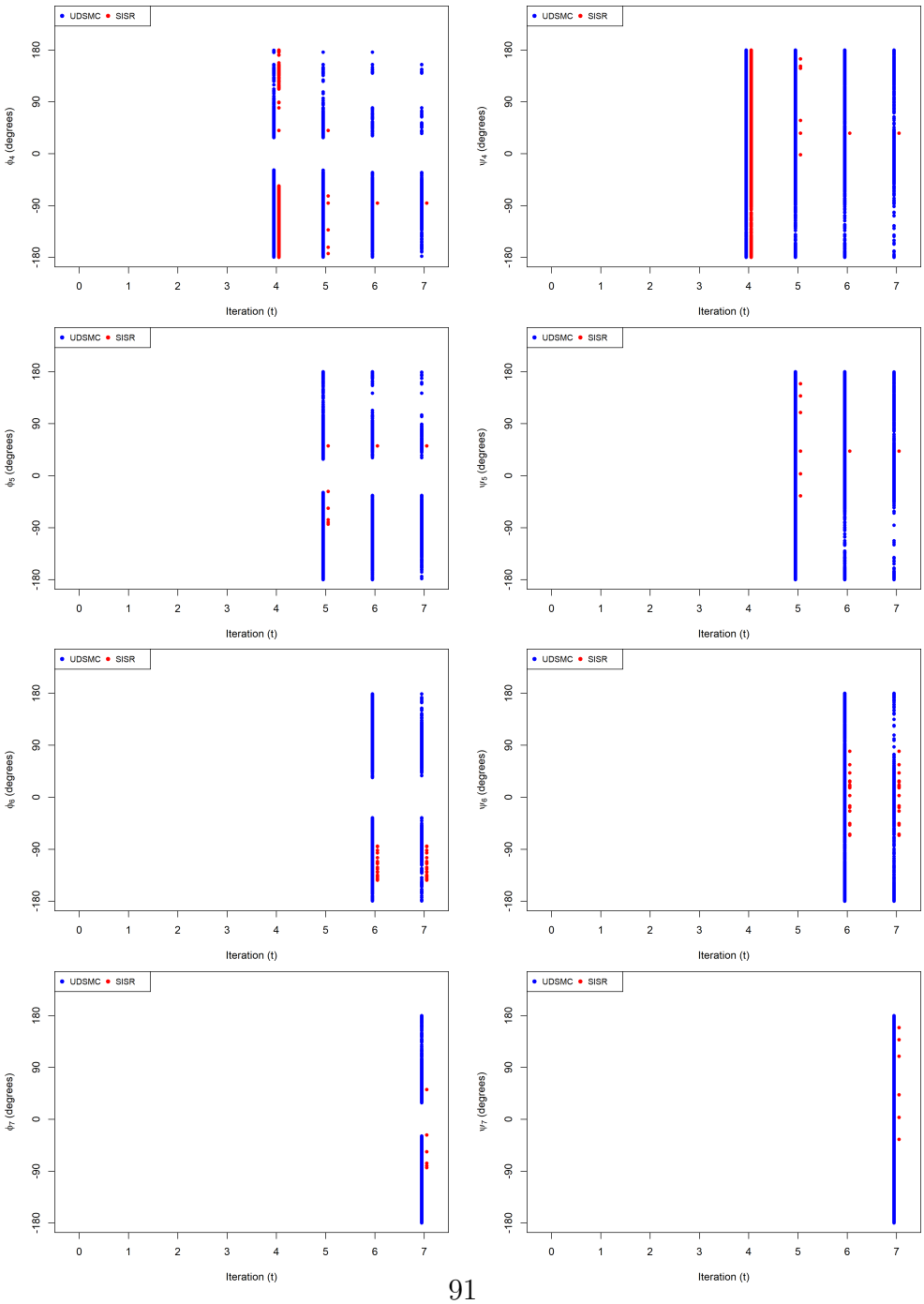


Figure A.2: Values of ϕ_s (left panels) and ψ_s (right panels) among resampled particles, for $s = 4, 5, 6, 7$ after each iteration (t) of UDSMC (blue) and SISR (red), for the protein segment in the simulation study.



Appendix B

Supplementary Material for Chapter 3

B.1 Proof of Proposition 3

In the ABC setting, we extend the dimension of the target distribution by introducing a sequence of auxiliary observations $u_{1:T}$. For simplicity, write $x_0 = h_0$ and $x_t = (h_t, u_t)$ for $t = 1, \ldots, T$, and we denote the target distribution by $p(x_{0:T}, \theta \mid r_{1:T})$ (dropping ϵ to simplify notation as it is known and fixed). Then we have the same target distribution as Andrieu et al. (2010), where they showed that particle Gibbs with the conditional bootstrap filter (cBF), i.e., Algorithm 1 in the main text, admits the target distribution as the invariant distribution under some mild assumptions (i.e., Assumptions 5-7 in Andrieu et al. (2010)). Here, we shall show that Algorithm 5 in the main text also admits the target distribution as the invariant distribution under the same mild assumptions.

The sweep of particle Gibbs has three steps (see Section 4.5 in Andrieu et al. (2010)): for cBF, given an input trajectory $x^*_{0:T}$,

1. update the parameters, denoted by θ^*
2. generate new trajectories with ancestor indices conditional on $x^*_{0:T}$ and θ^* using cBF
3. draw a trajectory from the union of the generated trajectories and the input trajectory, as the new input trajectory for the next iteration

and we propose to substitute the cBF for Step 2 with the conditional auxiliary particle filter (cAPF). Therefore, to prove the proposition, it is sufficient to show that with the same input trajectory, the cBF and the cAPF target the same distribution.

Write the target distribution of the cBF at each t as $\pi_t(x_{0:t} \mid \theta, r_{1:t})$ for $t = 1, \ldots, T$ and $\pi_0 = g_0(x_0 \mid \theta)$ (see Equation (33) in Andrieu et al. (2010) for the explicit form); for a particle $x_{0:t}$, its particle weight generated by cBF (Algorithm 1) is denoted by $w_t^{(\mathrm{cBF})}(x_{0:t})$, and its particle weight generated by cAPF (Algorithm 4) is denoted by $w_t^{(\mathrm{cAPF})}(x_{0:t})$. Then for the input trajectory $x_{0:t}^*$, it is obvious that we have $w_t^{(\mathrm{cBF})}(x_{0:t}^*) = w_t^{(\mathrm{cAPF})}(x_{0:t}^*)$. The input trajectory is set to be the N-th particle, i.e., $x_{0:t}^{(N)} = x_{0:t}^*$, for each t. Furthermore, for each generated trajectory $x_{0:t}^{(n)}$, $n = 1, \ldots, N-1$, we have

$$\begin{aligned} E\left\{w_t^{(\mathrm{cAPF})}(x_{0:t}^{(n)})h(x_{0:t}^{(n)})\right\} &= \int w_t^{(\mathrm{cAPF})}(x_{0:t}^{(n)})h(x_{0:t}^{(n)})\eta_t(x_{0:t}^{(n)})dx_{0:t}^{(n)} \\ &= \int \frac{w_t^{(\mathrm{cBF})}(x_{0:t}^{(n)})h(x_{0:t}^{(n)})}{\eta_t(x_{0:t}^{(n)})}\eta_t(x_{0:t}^{(n)})dx_{0:t}^{(n)} \\ &= E\left\{w_t^{(\mathrm{cBF})}(x_{0:t}^{(n)})h(x_{0:t}^{(n)})\right\} \end{aligned} \tag{B.1}$$

where h is any squared integrable function and $\eta_t(x_{0:t}^{(n)}) = \prod_{s=0}^{t-1} \widetilde{w}_s^{(A_{s,t}^{(n)})}$ with $\widetilde{w}_s^{(A_{s,t}^{(n)})}$ denoting the tempered weights in cAPF as defined in Equation (5) in the main paper. Given the same input trajectory, Equation (B.1) justifies that the particles produced by cBF and the particles produced by cAPF are properly weighted with respect to the same distribution $\pi_t(x_{0:t} \mid \theta, r_{1:t})$ (Liu, 2001), and thus these two algorithms target the same distribution at each step t.

B.2 Impact of a smaller ϵ in the ABC kernel

To further investigate the influence of ϵ in the ABC kernel, we set $\epsilon = 0.0005$ while keeping the rest of the settings to be the same as in the Simulation Study of the main paper. The results for the three algorithms, reported via parameter RMSEs analogous to Tables 1–3 of the main paper, are shown in Tables B.1, B.2 and B.3. A comparison of these results with $\epsilon = 0.001$ indicates that all three algorithms overall have higher RMSEs with this smaller ϵ, but ABC-PG-cAPF still consistently outperforms the other two algorithms. Moreover, these results suggest that weight degeneracy is more severe for ABC-PG-cBF and ABC-PG-cBFAS with the smaller ϵ, as their RMSEs increased significantly more than

those of ABC-PG-cAPF. Therefore, these results provide further numerical evidence that the proposed cAPF-based algorithm is more robust to weight degeneracy.

CV	ϕ	Algorithm	RMSE τ	RMSE ϕ	RMSE σ_h^2
10	0.9	ABC-PG-cBF	0.740	0.098	0.598
		ABC-PG-cBFAS	0.874	0.114	0.691
		ABC-PG-cAPF	0.214	0.037	0.163
10	0.95	ABC-PG-cBF	0.901	0.121	0.596
		ABC-PG-cBFAS	1.050	0.139	0.690
		ABC-PG-cAPF	0.447	0.068	0.201
10	0.98	ABC-PG-cBF	1.095	0.150	0.601
		ABC-PG-cBFAS	1.225	0.169	0.664
		ABC-PG-cAPF	0.691	0.103	0.298
1	0.9	ABC-PG-cBF	0.824	0.116	0.582
		ABC-PG-cBFAS	0.994	0.140	0.672
		ABC-PG-cAPF	0.328	0.054	0.264
1	0.95	ABC-PG-cBF	1.078	0.153	0.603
		ABC-PG-cBFAS	1.262	0.179	0.693
		ABC-PG-cAPF	0.577	0.089	0.285
1	0.98	ABC-PG-cBF	1.239	0.178	0.597
		ABC-PG-cBFAS	1.446	0.207	0.682
		ABC-PG-cAPF	0.818	0.125	0.330
0.1	0.9	ABC-PG-cBF	0.851	0.123	0.604
		ABC-PG-cBFAS	1.008	0.146	0.672
		ABC-PG-cAPF	0.379	0.061	0.345
0.1	0.95	ABC-PG-cBF	1.114	0.161	0.593
		ABC-PG-cBFAS	1.306	0.189	0.674
		ABC-PG-cAPF	0.682	0.105	0.356
0.1	0.98	ABC-PG-cBF	1.295	0.188	0.593
		ABC-PG-cBFAS	1.511	0.219	0.678
		ABC-PG-cAPF	0.892	0.136	0.370

Table B.1: RMSEs of the SVM parameter estimates for the three particle Gibbs algorithms, based on 100 simulated datasets with $T = 100$, $N = 100$, $Z_t \sim SD(1.75, 0.1, 1, 0)$, $\epsilon = 0.0005$.

CV	ϕ	Algorithm	RMSE τ	RMSE ϕ	RMSE σ_h^2
10	0.9	ABC-PG-cBF	0.765	0.101	0.620
		ABC-PG-cBFAS	0.898	0.118	0.710
		ABC-PG-cAPF	0.224	0.038	0.162
10	0.95	ABC-PG-cBF	0.942	0.126	0.635
		ABC-PG-cBFAS	1.056	0.140	0.700
		ABC-PG-cAPF	0.457	0.069	0.219
10	0.98	ABC-PG-cBF	1.091	0.151	0.596
		ABC-PG-cBFAS	1.263	0.174	0.694
		ABC-PG-cAPF	0.695	0.103	0.292
1	0.9	ABC-PG-cBF	0.875	0.123	0.623
		ABC-PG-cBFAS	1.022	0.145	0.690
		ABC-PG-cAPF	0.308	0.052	0.257
1	0.95	ABC-PG-cBF	1.090	0.155	0.616
		ABC-PG-cBFAS	1.288	0.182	0.705
		ABC-PG-cAPF	0.602	0.094	0.302
1	0.98	ABC-PG-cBF	1.271	0.183	0.607
		ABC-PG-cBFAS	1.491	0.214	0.715
		ABC-PG-cAPF	0.840	0.128	0.351
0.1	0.9	ABC-PG-cBF	0.870	0.126	0.618
		ABC-PG-cBFAS	1.024	0.149	0.693
		ABC-PG-cAPF	0.346	0.058	0.336
0.1	0.95	ABC-PG-cBF	1.125	0.163	0.607
		ABC-PG-cBFAS	1.348	0.195	0.706
		ABC-PG-cAPF	0.682	0.106	0.357
0.1	0.98	ABC-PG-cBF	1.325	0.192	0.619
		ABC-PG-cBFAS	1.555	0.225	0.715
		ABC-PG-cAPF	0.889	0.136	0.367

Table B.2: RMSEs of the SVM parameter estimates for the three particle Gibbs algorithms, based on 100 simulated datasets with $T = 100$, $N = 100$, $Z_t \sim SD(1.7, 0.3, 1, 0)$, $\epsilon = 0.0005$.

CV	ϕ	Algorithm	RMSE τ	RMSE ϕ	RMSE σ_h^2
10	0.9	ABC-PG-cBF	0.946	0.122	0.824
		ABC-PG-cBFAS	1.094	0.141	0.907
		ABC-PG-cAPF	0.258	0.044	0.197
10	0.95	ABC-PG-cBF	1.109	0.146	0.790
		ABC-PG-cBFAS	1.279	0.168	0.896
		ABC-PG-cAPF	0.503	0.076	0.243
10	0.98	ABC-PG-cBF	1.273	0.174	0.749
		ABC-PG-cBFAS	1.476	0.200	0.870
		ABC-PG-cAPF	0.730	0.109	0.318
1	0.9	ABC-PG-cBF	1.042	0.147	0.772
		ABC-PG-cBFAS	1.203	0.170	0.861
		ABC-PG-cAPF	0.350	0.060	0.298
1	0.95	ABC-PG-cBF	1.260	0.178	0.765
		ABC-PG-cBFAS	1.504	0.213	0.890
		ABC-PG-cAPF	0.655	0.102	0.346
1	0.98	ABC-PG-cBF	1.430	0.205	0.753
		ABC-PG-cBFAS	1.686	0.241	0.878
		ABC-PG-cAPF	0.880	0.135	0.382
0.1	0.9	ABC-PG-cBF	1.038	0.150	0.764
		ABC-PG-cBFAS	1.230	0.179	0.862
		ABC-PG-cAPF	0.419	0.069	0.389
0.1	0.95	ABC-PG-cBF	1.264	0.183	0.727
		ABC-PG-cBFAS	1.546	0.224	0.867
		ABC-PG-cAPF	0.712	0.112	0.389
0.1	0.98	ABC-PG-cBF	1.483	0.215	0.751
		ABC-PG-cBFAS	1.733	0.252	0.860
		ABC-PG-cAPF	0.944	0.145	0.420

Table B.3: RMSEs of the SVM parameter estimates for the three particle Gibbs algorithms, based on 100 simulated datasets with $T = 100$, $N = 100$, $Z_t \sim SD(1.5, -0.3, 1, 0)$, $\epsilon = 0.0005$.

Appendix C

Supplementary Material for Chapter 4

C.1 Trajectory plots generated with one of the 100 estimates

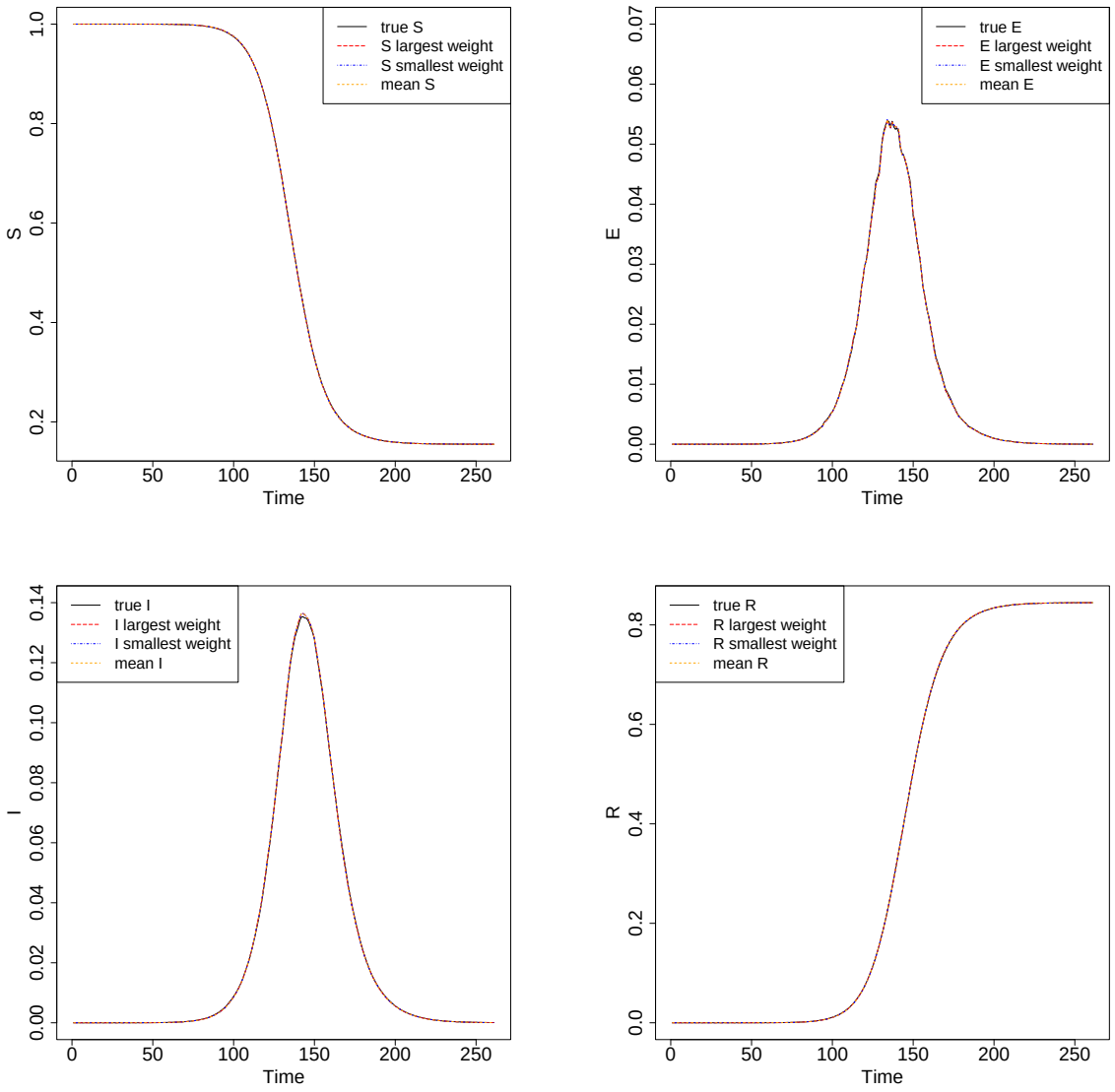


Figure C.1: The four figures demonstrate the simulated trajectories of the four compartments using the kernel H defined in (i) in Section 4.3 with respect to a set of randomly-picked estimates for the parameters from the 100 repetitions. Three trajectories simulated from the ABC-SMC using the estimates are presented in each figure, including the one with the largest particle weight, the one with the smallest particle weight, and the mean trajectory.

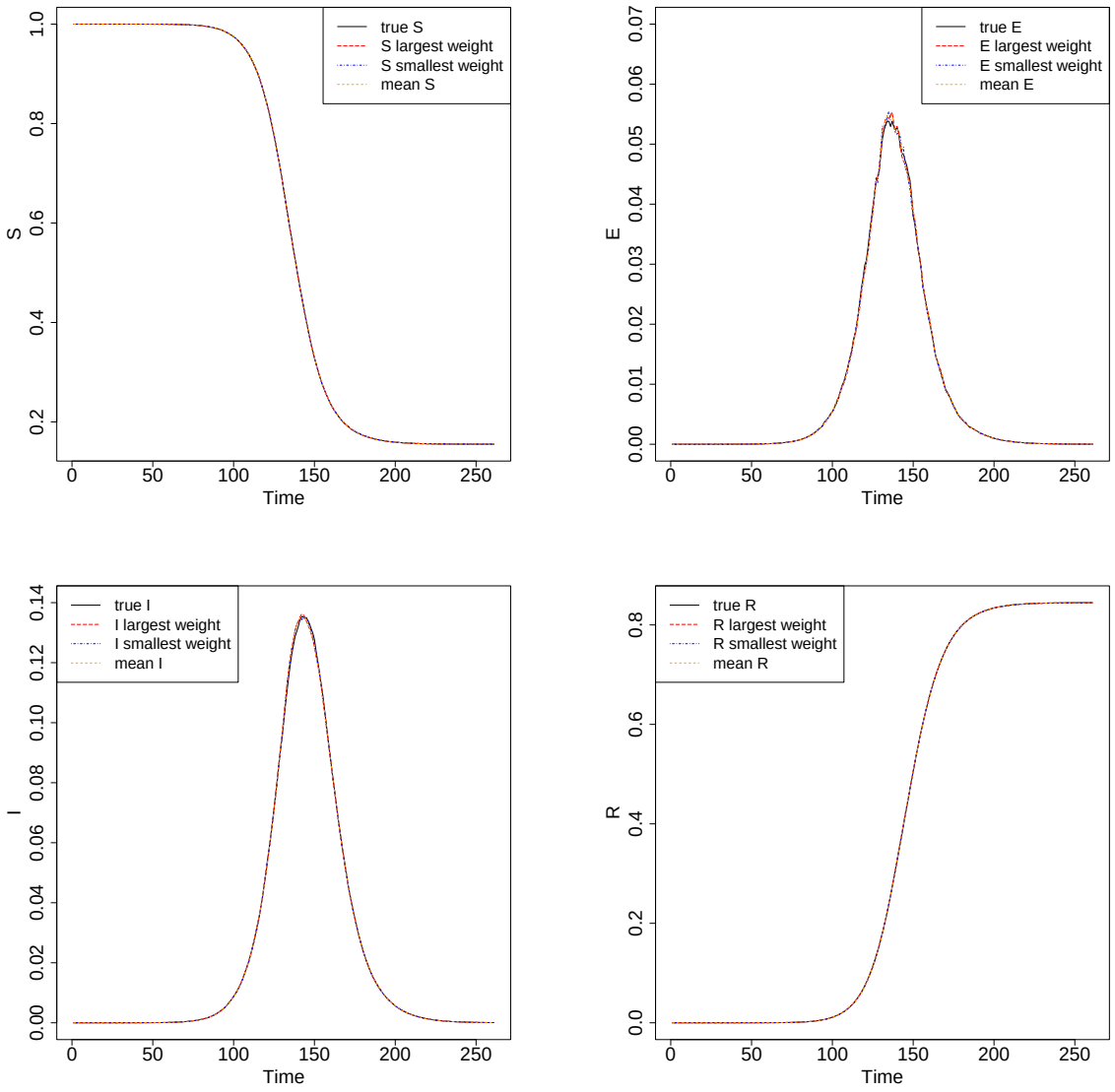


Figure C.2: The four figures demonstrate the simulated trajectories of the four compartments using the kernel H defined in (ii) in Section 4.3 with respect to a set of randomly-picked estimates for the parameters from the 100 repetitions. Three trajectories simulated from the ABC-SMC using the estimates are presented in each figure, including the one with the largest particle weight, the one with the smallest particle weight, and the mean trajectory.

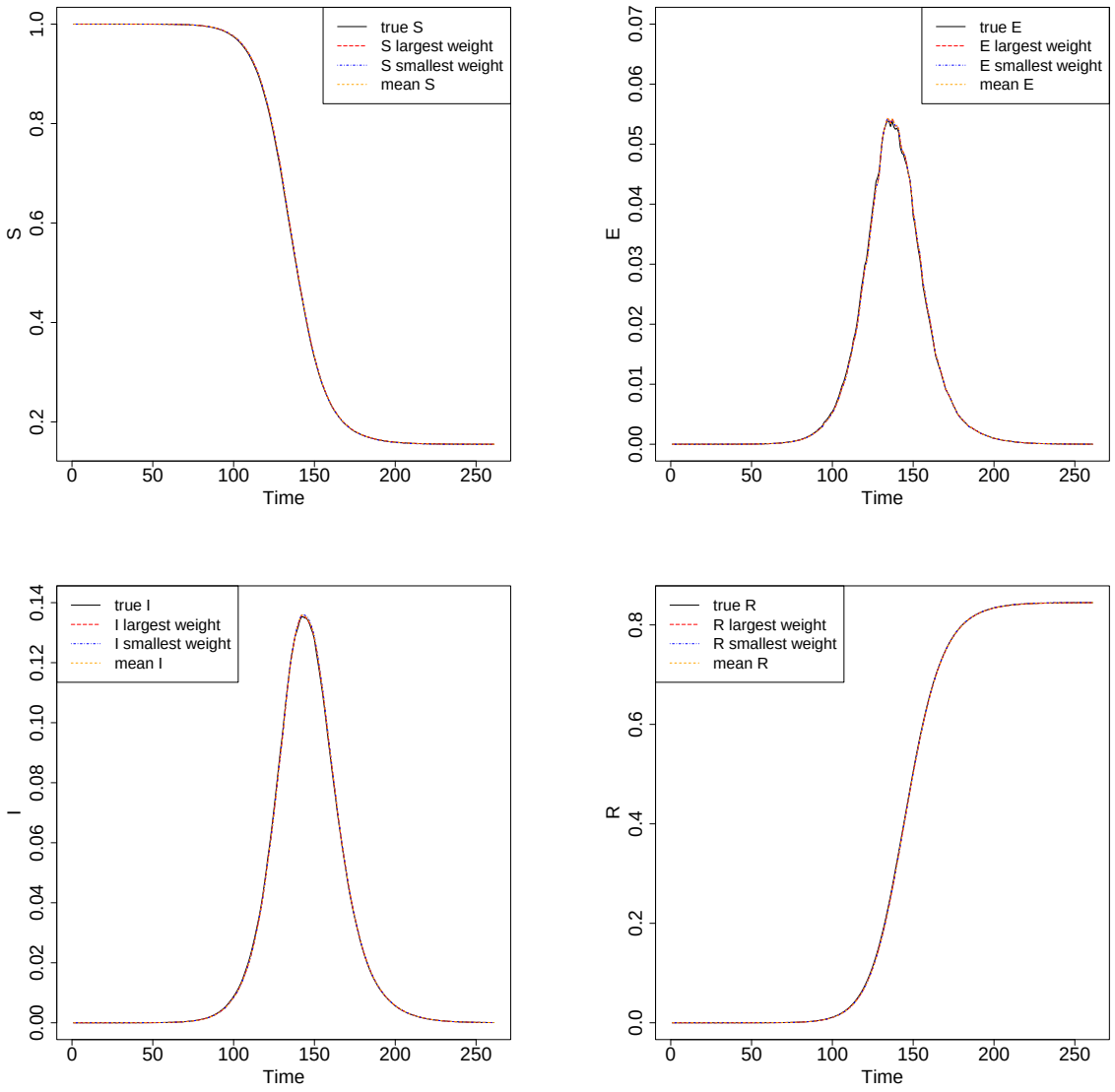


Figure C.3: The four figures demonstrate the simulated trajectories of the four compartments using the kernel H defined in (iii) in Section 4.3 with respect to a set of randomly-picked estimates for the parameters from the 100 repetitions. Three trajectories simulated from the ABC-SMC using the estimates are presented in each figure, including the one with the largest particle weight, the one with the smallest particle weight, and the mean trajectory.