Monte Carlo Methods with Space Domain Applications

by

Caleb Miller

B.S., California Polytechnic State University, San Luis Obispo 2016

M.S., California Polytechnic State University, San Luis Obispo, 2016

M.S., University of Colorado, Boulder, 2018

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy in Applied Mathematics

Department of Applied Mathematics University of Colorado, Boulder

2021

Committee Members:

Dr. Jem N. Corcoran, Chair

Dr. Michael D. Schneider

Dr. Ian Grooms

Dr. Jim Curry

Dr. Tomoko Matsuo
Monte Carlo methods utilize random samples to provide numerical estimates, and are used in many application areas. Naive implementation of Monte Carlo algorithms are often computationally wasteful, especially in the case of rare event estimation. A popular tool for Monte Carlo estimation are Markov Chain Monte Carlo (MCMC) algorithms. The methods used in this thesis opt instead for utilizing the theory of perfect sampling and importance sampling which are free from the usually considerations of MCMC methods such as burn-in time, mixing-time, and correlation of samples.

We leverage the concept of perfect simulation, obtaining draws directly from stationary distributions, to sample from the distribution of Non-IID Ordered Random Variates. This sampling allows us to construct a perfect simulation algorithm for Bayesian principle component analysis. We demonstrate the effectiveness of the algorithm on several examples.

The remaining work in the thesis was inspired by data fusion and conjunction analysis problems in the space domain. Given the task of fusing samples drawn from partial posterior distributions we utilize ideas from particle filtering and multiple importance sampling to fuse the estimates via the cross-pollination algorithm. We prove the convergence of the algorithm in the particle limit and demonstrate efficacy on fusing samples to estimate the parameters of a Keplerian orbit via fusion of different samples obtained at different times and locations. Finally, we present a novel algorithm for estimation of rare events. We reinterpret the typical cross-entropy derivation with the deterministic mixture weights of multiple importance sampling to realize a natural updating formula for a population Monte Carlo scheme. We demonstrate this method on several rare event problems and demonstrate efficacy on a conjunction analysis problem.
Dedication

Thank you to my family for your support and constant ability to make me laugh. Thank you to my friends for roughly the same reason. Thank you to an excellent Calculus teacher, Ryan Pedersen, who I asked one day what it would be like to be a math major, his response thrilled me enough to begin this journey. Thank you to Dr. Benjamin Richert for advising during my undergraduate years. Thank you to each member of my committee: Dr. Corcoran introduced me to stochastic simulation and was an excellent advisor, Dr. Schneider introduced me to problems in many areas (only some of which appear here), Dr. Grooms pointed me in the right direction when attacking problems more than once, Dr. Curry advised me on more than mathematics, and Dr. Matsuo provided me with my first research opportunities in Boulder.
Acknowledgements

I thank Ian Grooms for discussions that aided Chapter 3. I thank Víctor Elvira for helpful comments that aided Chapter 4. Chapters 3 and 4 were performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. Funding for this work was provided under LLNL Laboratory Directed Research and Development 19-SI-004.
## Contents

### Chapter

1. **A Brief Background**
   1.1 Estimating Moments of a Probability Distribution Function .......................... 1
   1.2 Importance Sampling ................................................................................. 2
   1.3 Perfect Sampling ....................................................................................... 3
   1.4 Space Domain Applications: Data Fusion and Conjunction Analysis ............. 3

2. **$\varepsilon$-Perfect Sampling for Non-IID Order Statistics** .......................... 5
   2.1 Contribution ............................................................................................ 5
   2.2 Introduction ............................................................................................. 5
   2.3 The Perfect Gibbs Sampler ....................................................................... 6
      2.3.1 Bounding Processes ......................................................................... 8
      2.3.2 The Algorithm ................................................................................ 11
   2.4 Examples .................................................................................................. 11
      2.4.1 Exponential Distributions ................................................................. 11
      2.4.2 Weibull Distributions ...................................................................... 13
      2.4.3 Cauchy Distributions ...................................................................... 16
      2.4.4 Pareto Distributions ......................................................................... 16
      2.4.5 Exponential, Weibull, and a “Folded” Cauchy Distribution ............... 16
      2.4.6 Bayesian Principal Components Analysis ........................................ 20
   2.5 Conclusion ............................................................................................... 24
<table>
<thead>
<tr>
<th>3  Cross-Pollination</th>
<th>26</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1 Contribution</td>
<td>26</td>
</tr>
<tr>
<td>3.2 Introduction</td>
<td>27</td>
</tr>
<tr>
<td>3.3 Method</td>
<td>27</td>
</tr>
<tr>
<td>3.3.1 Objective, Set-Up, and Requirements</td>
<td>27</td>
</tr>
<tr>
<td>3.3.2 Measure Theoretic Description of Method</td>
<td>28</td>
</tr>
<tr>
<td>3.3.3 Relation to Multiple Importance Sampling</td>
<td>31</td>
</tr>
<tr>
<td>3.3.4 Proof of Method</td>
<td>32</td>
</tr>
<tr>
<td>3.4 Other Methods</td>
<td>35</td>
</tr>
<tr>
<td>3.5 Examples</td>
<td>36</td>
</tr>
<tr>
<td>3.5.1 Convergence Verification and Comparison on Gamma Example</td>
<td>36</td>
</tr>
<tr>
<td>3.5.2 Multiple Sensor Bearings</td>
<td>40</td>
</tr>
<tr>
<td>3.5.3 Orbit Determination</td>
<td>42</td>
</tr>
<tr>
<td>3.5.4 Multiple Sensor Sequential Bearings</td>
<td>44</td>
</tr>
<tr>
<td>3.6 Discussion</td>
<td>47</td>
</tr>
<tr>
<td>3.7 Algorithms</td>
<td>49</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>4  Cross Entropy Population Monte Carlo</th>
<th>51</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1 Contribution</td>
<td>51</td>
</tr>
<tr>
<td>4.2 Introduction</td>
<td>52</td>
</tr>
<tr>
<td>4.3 Rare Events and Importance Sampling</td>
<td>53</td>
</tr>
<tr>
<td>4.3.1 Rare Event Problem</td>
<td>53</td>
</tr>
<tr>
<td>4.3.2 Importance Sampling</td>
<td>53</td>
</tr>
<tr>
<td>4.4 Cross-Entropy Method</td>
<td>55</td>
</tr>
<tr>
<td>4.5 Cross-Entropy Population Monte Carlo</td>
<td>56</td>
</tr>
<tr>
<td>4.6 Examples</td>
<td>59</td>
</tr>
<tr>
<td>4.6.1 Structural Reliability Examples</td>
<td>60</td>
</tr>
</tbody>
</table>
4.6.2 Variable Dimension Problem ............................................. 62
4.6.3 Conjunction Analysis .................................................... 63
4.6.4 Revisiting Data Fusion ................................................... 65
4.7 Conclusions and Future Directions ..................................... 66

Bibliography ................................................................. 68
# Tables

## Table

2.1  Mean backward coupling times for the exponential, Weibull, Cauchy, and Pareto distributions  
3.1  RMSE on Orbit Samples 
4.1  RRMSE on Structural Reliability Examples
## Figures

**Figure**

<table>
<thead>
<tr>
<th>Figure</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Updating paths for backward coupling</td>
<td>8</td>
</tr>
<tr>
<td>2.2</td>
<td>Simulation results for the exponential distribution</td>
<td>14</td>
</tr>
<tr>
<td>2.3</td>
<td>Simulation results for the Weibull distribution</td>
<td>15</td>
</tr>
<tr>
<td>2.4</td>
<td>Simulation results for the Cauchy distribution</td>
<td>17</td>
</tr>
<tr>
<td>2.5</td>
<td>Simulation results for the Pareto distribution</td>
<td>18</td>
</tr>
<tr>
<td>2.6</td>
<td>Simulation results with different base distributions</td>
<td>19</td>
</tr>
<tr>
<td>2.7</td>
<td>Scoring the models</td>
<td>24</td>
</tr>
<tr>
<td>3.1</td>
<td>A schematic for the Cross-Pollination Process</td>
<td>31</td>
</tr>
<tr>
<td>3.2</td>
<td>Convergence of Cross-Pollination on gamma example</td>
<td>38</td>
</tr>
<tr>
<td>3.3</td>
<td>Cross-Pollination and competing methods on gamma example</td>
<td>39</td>
</tr>
<tr>
<td>3.4</td>
<td>Samples at different stages on bearing example</td>
<td>41</td>
</tr>
<tr>
<td>3.5</td>
<td>Samples at different stages on orbit example</td>
<td>43</td>
</tr>
<tr>
<td>3.6</td>
<td>Samples at different epochs and refinement for sequential cross-pollination</td>
<td>47</td>
</tr>
<tr>
<td>4.1</td>
<td>CE-PMC contour plots on different regions of importance</td>
<td>61</td>
</tr>
<tr>
<td>4.2</td>
<td>Mean probability of LR-PMC, GR-PMC, and CE-PMC on variable dimension problem</td>
<td>62</td>
</tr>
<tr>
<td>4.3</td>
<td>Traces of orbital positions with naive Monte Carlo samples and refined CE-PMC samples</td>
<td>64</td>
</tr>
<tr>
<td>4.4</td>
<td>Traces of true orbital positions and CE-PMC samples</td>
<td>66</td>
</tr>
</tbody>
</table>
Chapter 1

A Brief Background

1.1 Estimating Moments of a Probability Distribution Function

The expectation of a moment of a probability distribution function is paramount to estimation. Consider a discrete random variable $X$ with outcomes $x_1, \ldots, x_N$ with associated probabilities $p_1, \ldots, p_N$. The expected value is $E[X] = \sum_{i=1}^{N} x_i \cdot p_i$. For a continuous random variable with probability distribution function $p(x)$ the expectation is given by $\ell = E_p[X] = \int x \cdot p(x) \, dx$. Furthermore, when one is interested in expectations of functions of random variables we have the law of the unconscious statistician that is

$$E_p[h(x)] = \int h(x) \cdot p(x) \, dx.$$  

To gain information from these expressions we are often forced to do some estimation. In a class on integral calculus one would build up quadrature rules to estimate these integrals. We are concerned with the Monte Carlo methods which differ from quadrature in that instead of gridding space in some way Monte Carlo methods work by drawing samples from probability distribution functions and then constructing estimators. To wit, the naive Monte Carlo estimator of the expectation is

$$\hat{\ell}_{MC} = \frac{1}{K} \sum_{k=1}^{K} h(x_k) \quad \text{where} \quad x_k \sim p(x).$$

Which is interpreted as drawing $K$ samples from the pdf $p$ and then taking their average after applying the function $h$. The Strong Law of Large Numbers provides proof as to why this is a desirable estimator in an almost surely sense [52, Sec 3.2]. An alternative approach considered by Law et al. [36, Sec 4.3] in their proofs regarding particle filters is to consider a sampling operator, which operates on the space of probability
measures. Here the distance considered is related to the total variation norm and is

\[ d(\mu, \nu) = \sup_{|f| \leq 1} \sqrt{\mathbb{E}|\mu(f) - \nu(f)|}, \]

where \( \mu(f) = \int f(v)\mu(dv) \). And the sampling operator is defined by

\[ (S^K\mu)(dv) = \frac{1}{K} \sum_{k=1}^{K} \delta_{x_k}(dx), \quad x_k \sim \mu \]

The \( \delta \) are dirac measures located at the location of the samples. Notice that to return to the Monte Carlo estimator we would apply \( h \) to the location of the particles provided by \( \mu \) if \( \mu \) was the measure that corresponded to the pdf \( p \). The result of the proof in Law et al. was that \( d(S^K\mu, \mu) \leq \frac{1}{\sqrt{K}} \) which is another interpretation of the common convergence of sampling being on the order of the square root of the number of samples, this time under an alternative mode of convergence. This becomes a vital tool in proving the convergence of our method in Chapter 3.

### 1.2 Importance Sampling

A ubiquitous tool for improving standard Monte Carlo methods is that of importance sampling. The jist of importance sampling is to use an alternative distribution \( q \) to propose samples from. Importance sampling attacks several problems with the standard Monte Carlo estimator including: difficulty in sampling from the original distribution (whether that is in the production of a sample or even having an analytic form of the original distribution) and improving efficiency of the sampling by finding important regions of uncertainty. To see the initial simplicity of importance sampling notice

\[ \mathbb{E}_p[h(x)] = \int h(x)p(x)\,dx = \int h(x)\frac{p(x)}{q(x)}\,dx = \mathbb{E}_q \left[ h(x)\frac{p(x)}{q(x)} \right]. \]

Thus the importance sampling estimator is given by

\[ \hat{\ell}_{IS} = \frac{1}{K} \sum_{k=1}^{K} h(x_k)\frac{p(x_k)}{q(x_k)}, \quad x_k \sim q(x). \]

The expressions \( p(x_k)/q(x_k) \) are known as the importance weights for the samples. These weights are what match the expectations when the sampling distribution is changed and can be interpreted as the Radon Nikodym derivative between the measures corresponding to the probability density functions \( p \) and
Naturally, advances in importance sampling consider using multiple proposal distributions and adapting proposal distributions through trials. For further reading on the subject we refer the reader to [52], [17] and [8].

### 1.3 Perfect Sampling

A Markov chain Monte Carlo (MCMC) algorithm allows us to create a Markov chain that converges to a draw from a given target distribution. Perfect simulation (sampling) algorithms comprise a subclass of MCMC algorithms that are completely free of convergence error. The idea of perfect simulation is to find a random time $-T$, known as a backward coupling time (BCT), in the past such that, if we construct every possible sample path for the chain starting at time $-T$, all paths will have come together or “coupled” by time zero. The common value of the paths at time zero is a draw from the limiting distribution for the chain. Intuitively, it is clear why this result holds. Consider a sample path starting at time $-\infty$. When this path reaches time $-T$ it must pick some value $x$, and from then on it follows the trajectory from that value. By construction of $T$, it arrives at the same place at time zero no matter what value $x$ is picked at time $-T$, so the value returned by the algorithm at time zero is the tail end of a sample path that has run for an infinitely long time. Construction of “every possible sample path” is usually facilitated by exploiting a monotonicity structure which will allow us to consider only two paths, one from the top of the space and one from the bottom, that will sandwich all possible paths in between. If the space is unbounded, “stochastically dominating” processes can be used to create upper and lower bounds for the process of interest. For further details on perfect simulation in general, we refer the reader to [11], [12], [22], [48], and [51].

### 1.4 Space Domain Applications: Data Fusion and Conjunction Analysis

The work in Chapter 3 and 4 of this thesis was motivated by the need to create accurate uncertainty quantification algorithms in the areas of data fusion and conjunction analysis.

The first problem posed was that of the data fusion problem. If we had batches of estimates of states or parameters of objects, perhaps these batches were based on noisy observations, or observations through different modalities such as optical and radar measurements. How could the batches of estimates be fused
to get a refined idea or picture of the scenario. Specifically, we were interested in the problem of orbit
determination. Schneider et al. [56] demonstrated the first use of the cross-pollination algorithm. Working
together with Schneider, I refined this method for greater accuracy and prove its convergence in Chapter 3.

The second problem was that of conjunction analysis. There are many different facets of this problem,
put simply, what is the probability that two objects in space come within some distance of each other. Many
assumptions can be made to simplify the problem, for instance a short-term encounter with linear Gaussian
assumptions can be solved with characteristic function inversion [6]. Less ideal scenarios require the use
of Monte Carlo methods. For the scenario in which two objects involved both have uncertainty and linear
Gaussian assumptions can’t be justified, Monte Carlo methods must be used for estimates. The most popular
methods are that of the each-to-each and all-to-all estimators, these methods are well explained and advances
in the all-to-all method are presented in Bernstein et al. [5]. The conjunction problem at its core is a rare
event problem– it is unlikely for a satellite to crash into another. The popular all-to-all and each-to-each
methods are at essentially two versions of naive Monte Carlo methods, therefore they are wasteful and more
advanced methods must be created to reduce computational waste. An advance in this area was the use
of subset simulation an MCMC method for rare events by Losacco et al. [39]. We alternatively propose an
adaptive importance sampling methodology and show its ability to be useful in conjunction analysis problems
in Chapter 4.
Chapter 2

ε-Perfect Sampling for Non-IID Order Statistics

This chapter presents a paper Prof. Jem Corcoran and I wrote, which can be found at https://arxiv.org/abs/2012.15452.

2.1 Contribution

In this chapter, we consider the problem of simulating ordered values \(X_1, X_2, \ldots, X_m\) such that the marginal distribution of \(X_i\) is \(F_i(x)\). This problem arises in Bayesian principal components analysis (BPCA) where the \(X_i\) are ordered eigenvalues that are a posteriori independent but not identically distributed. In this chapter, we introduce a novel coupling-from-the-past algorithm to “perfectly” (up to computable order of accuracy) simulate such order-constrained non-iid order statistics. We demonstrate the effectiveness of our approach for several examples, including the BPCA problem.

2.2 Introduction

Suppose that \(X_1, X_2, \ldots, X_m\) are independent and identically distributed (iid) random variables from a distribution with probability density function (pdf) \(f\). The order statistics are defined and denoted as \(X_1, X_2, \ldots, X_m\) where \(X_1 = \min(X_1, X_2, \ldots, X_m)\), \(X_m = \max(X_1, X_2, \ldots, X_m)\), and, in general, \(X_i\) is the \(i\)th smallest value from the sample \(X_1, X_2, \ldots, X_m\). Assuming that one can simulate values from the distribution with pdf \(f\), one can simulate order statistics by simulating \(m\) iid values from \(f\) and then simply putting them in order. In the case of independent and non-identically distributed (inid) random variables, where \(X_i\) comes from a distribution with pdf \(f_i\), order statistics \(X_1, X_2, \ldots, X_m\) are still defined in the
same way. One can simulate these iid order statistics by independently simulating $X_i$ from the distribution with pdf $f_i$ and then putting them in order. In this case, the $i$th order statistic is not necessarily the value simulated from the $i$th distribution. Imposing such a constraint makes methods of simulation much more challenging and is the subject of this chapter. We will call such random variates order constrained independent and non-identically distributed (ocinid) order statistics.

The problem of simulating ordered values corresponding to specific non-identical distributions often comes up in the context of Bayesian principal components analysis (BPCA), where it is of interest to be able to simulate ordered eigenvalues from a posterior distribution \[34\], \[44\], \[59\], \[63\]. We will discuss this particular problem in more detail in Section 2.4.6.

In this chapter, we will use the idea of perfect simulation or coupling-from-the-past (CFTP) in order to simulate $X_1, X_2, \ldots, X_m$ from a joint density $f(x_1, x_2, \ldots, x_m)$ of the form

\[
f(x_1, x_2, \ldots, x_m) \propto \prod_{i=1}^{m} f_i(x_i) I_{\{x_1 < x_2 < \cdots < x_m\}}
\]

(2.1)

where $I_{\{x_1 < x_2 < \cdots < x_m\}}$ is an indicator taking the value 1 if $x_1 < x_2 < \cdots < x_m$ and zero otherwise. The $f_i$ are pdfs that, in BPCA, are typically from the same family of distributions, often differing through a scale parameter. However, for the algorithm we present in this chapter they need not be related in any way other than by having the same support. We will refer to the joint density given by (2.1) as our “target density” and, in this chapter, we introduce a novel “\(\varepsilon\)-perfect” algorithm for sampling from (2.1). Rather than achieving a complete coupling of relevant sample paths, the algorithm will ensure that they are arbitrarily close and guaranteed to stay that way. In theory, complete coupling can be achieved for these paths using, for example, a “slice sampler” such as Mira et al. \[45\], but there is little to be gained from this effort since we can take $\varepsilon$ smaller than machine precision.

### 2.3 The Perfect Gibbs Sampler

The Gibbs Sampler \[23\] is a popular MCMC technique used to sample from an $n$-dimensional distribution, for $n \geq 2$, using its conditional distributions. A Gibbs sampler targeting (2.1) assumes that one can
sample directly from the conditional densities

\[ f(x_i|x_{-i}) \propto f_i(x_i) I_{x_{i-1}<x_i<x_{i+1}} \]  

(2.2)

do the target density (2.1) for \( i = 1, 2, \ldots, m \). Here, \( x_{-i} \) is the \((m-1)\)-dimensional vector \((x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_m)\). If we assume that \( f_i(x) \) has an associated invertible cdf \( F_i(x) \), we can sample from truncated version of \( f_i \) in (2.2) by sampling \( U \sim \text{Uniform}(F_i(x_{i-1}), F_i(x_{i+1})) \) and returning \( F_i^{-1}(U) \).

A typical non-perfect forward Gibbs sampler targeting (2.1) starts at some arbitrary point \((X_1^{(0)}, X_2^{(0)}, \ldots, X_m^{(0)})\) at time 0 and then moves between time steps \( n \) and \( n+1 \) as described in Algorithm 1.

**Algorithm 1 The Gibbs Sampler: A Move from Time \( n \) to Time \( n+1 \)**

**Input:** A starting value \((X_1^{(n)}, X_2^{(n)}, \ldots, X_m^{(n)})\)

Sample \( X_1^{(n+1)} \sim f(x_1|X_2^{(n)}, \ldots, X_m^{(n)}) \propto f_1(x_1) I_{-\infty<x_1<X_2^{(n)}} \)

for \( j = 2 \) to \( m-1 \) do

Sample \( X_j^{(n+1)} \sim f(x_j|X_1^{(n+1)}, \ldots, X_{j-1}^{(n+1)}, X_3^{(n)}, X_{j+1}^{(n)}, \ldots, X_m^{(n)}) \propto f_n(x_n) I_{X_{j-1}^{(n+1)}<x_n<X_j^{(n)}} \)

end

Sample \( X_m^{(n+1)} \sim f(x_m|X_1^{(n+1)}, \ldots, X_{m-1}^{(n+1)}) \propto f_m(x_m) I_{X_{m-1}^{(n+1)}<x_m<\infty} \).

Let \( V_{n+1} = (V_1^{(n+1)}, V_2^{(n+1)}, \ldots, V_m^{(n+1)}) \) be a vector of iid \( \text{Uniform}(0,1) \) random variables. If we define \( X_{-i}^{(n)} \) to be \((X_1^{(n)}, \ldots, X_{i-1}^{(n)}, X_{i+1}^{(n)}, \ldots, X_m^{(n)})\), we may describe the Gibbs update from time \( n \) to time \( n+1 \) with the following stochastic recursive sequence representation.

\[ X_i^{(n+1)} := \phi_i(X_{-i}^{(n)}, V_i^{(n+1)}) \]

\[ := \begin{cases}  
F_1^{-1}[F_1(X_2^{(n)}) \cdot V_1^{(n+1)}] & \text{, } i = 1 \\
F_i^{-1} \left[ F_i(X_{i-1}^{(n+1)}) + \left( F_i(X_{i+1}^{(n+1)}) - F_i(X_{i-1}^{(n+1)}) \right) V_i^{(n+1)} \right] & \text{, } i = 2, 3, \ldots, m-1 \\
F_m^{-1} \left[ F_m(X_{m-1}^{(n+1)}) + \left( 1 - F_m(X_{m-1}^{(n+1)}) \right) V_m^{(n+1)} \right] & \text{, } i = m.
\]

(2.3)

Note that, in all cases, we have that \( X_i^{(n+1)} \) is an increasing function of \((X_{i-1}^{(n+1)}, X_{i+1}^{(n+1)})\). This stochastic monotonicity will be a key observation to exploit for our backward coupling algorithm. Specifically, if
A single time step of a single sample path started at $X_{-n} = (X_{-n}^1, \ldots, X_{-n}^m)$ at some time $-n$ can be obtained through $m$ moves as depicted in Figure 2.1(a) in the case that $m = 4$. Here, move numbers are labeled on arrows. We want to achieve a “coupling” or “coalescence” of paths for all possible values of $X_i$ at time $-n$ by time 0, for each of $i = 1, 2, 3, 4$, into four distinct points $X_i^{(0)}$, $X_i^{(0)}$, $X_i^{(0)}$, and $X_i^{(0)}$ as depicted (in an overly idealized way) in Figure 2.1(b). Note that the lower paths, for example depicted in the “funnel” for $X_3$, are generated by simulating $X_3$ from the conditional density $f(x_3|x_1, x_2, x_4) = f(x_3|x_2, x_4)$ using values of $x_2$ from the lower paths of the $X_2$ funnel and values of $x_4$ from the lower paths of the $X_4$ funnel. Likewise, upper paths for $X_3$ are generated by simulating $X_3$ from $f(x_3|x_2, x_4)$ using upper paths from $X_2$ and $X_4$.

2.3.1 Bounding Processes

With monotonicity in our model, we will be able to capture the behavior of large collections of sample paths between paths from the “top” and “bottom” of the space for each of the $m$ components of our chain. If the $f_i$ have unbounded support, such points will not exist. However, with our model, we can actually run
upper and lower processes from $\infty$ and $-\infty$, respectively, using the fact that the involved cdfs are increasing and bounded between 0 and 1.

**An Upper Process**

Fix a time $-n$ and consider starting sample paths forward to time 0 from lower and upper points $X_{i,L}^{(-n)}$ and $X_{i,U}^{(-n)}$ for $i = 1, 2, \ldots, m$. Note that

$$X_{1,L}^{(-n)} = F_1^{-1}[F_1(X_{2,L}^{(-n-1)}) \cdot V_1^{(-n)}] \leq F_1^{-1}[V_1^{(-n)}].$$

So, we define the first component of the upper process started at time $-n$ as

$$X_{1,U}^{(-n)} = F_1^{-1}[V_1^{(-n)}] \quad (2.4)$$

which will be above $X_{1,L}^{(-n)}$ generated by any and all possibilities for $X_2$ at time $-n - 1$.

Similarly, for $i = 2, 3, \ldots, m - 1$, we have

$$X_{i,L}^{(-n)} = F_i^{-1}[F_i(X_{i+1,L}^{(-n-1)}) + (F_i(X_{i,L}^{(-n-1)}) - F_i(X_{i-1,L}^{(-n)})) \cdot V_i^{(-n)}] \leq F_i^{-1}[F_i(X_{i-1,U}^{(-n)}) + (1 - F_i(X_{i-1,L}^{(-n)})) \cdot V_i^{(-n)}].$$

Thus, we define

$$X_{i,U}^{(-n)} = F_i^{-1}[F_i(X_{i-1,U}^{(-n)}) + (1 - F_i(X_{i-1,L}^{(-n)})) \cdot V_i^{(-n)}] \quad (2.5)$$

which is an upper bound on $X_{i,L}^{(-n)}$.

Finally, we have that

$$X_{m,U}^{(-n)} := F_m^{-1}[F_m(X_{m-1,U}^{(-n)}) + (1 - F_m(X_{m-1,L}^{(-n)})) \cdot V_m^{(-n)}]$$

is an upper bound on

$$X_{m,U}^{(-n)} = F_m^{-1}[F_m(X_{m-1}^{(-n)}) + (1 - F_m(X_{m-1}^{(-n)})) \cdot V_m^{(-n+1)}]$$

for all finite $X_{m-1}^{(-n)}$.

In defining this upper process starting at time $n$, we have essentially started at time $-n - 1$ with $X_{i,U}^{(-n-1)} = \infty$ for all $i = 1, 2, \ldots, m$ and used the updates given in (2.3).
A Lower Process

The lower process is slightly more complicated. Using, for example, $X_{2,L}^{(-n-1)} = -\infty$ would result in an update to the first component of the lower process of

$$X_{1,L}^{(-n)} = F_1^{-1}[F_1(X_{2,L}^{(-n-1)})] = F_1^{-1}[0] = -\infty.$$ 

Indeed, all components of a lower process constructed in this manner would remain at $-\infty$ for all time. To avoid this, we will instead begin by bounding the largest (mth) component of the lower process and working our way down to the smallest component, while taking care to preserve the original update order. Note that

$$X_m^{(-n)} = F_m^{-1}[F_m(X_{m-1}^{(-n)}) + (1 - F_m(X_{m-1}^{(-n)})) \cdot V_m^{(-n)}]$$
$$\geq F_m^{-1}[F_m(-\infty) + (1 - F_m(-\infty)) \cdot V_m^{(-n)}] = F_m^{-1}[V_m^{(-n)}] =: X_{m,L}^{(-n)}$$

is a lower bound for $X_m^{(-n)}$ for all n. Similarly,

$$X_{m-1}^{(-n)} = F_{m-1}^{-1}[F_{m-1}(X_{m-2}^{(-n)}) + (F_{m-1}(X_{m-2}^{(-n-1)}) - F_{m-1}(X_{m-1}^{(-n-1)})) \cdot V_i^{(-n)}]$$
$$\geq F_{m-1}^{-1}[F_{m-1}(X_{m-2}^{(-n-1)}) \cdot V_i^{(-n)}] \geq F_{m-1}^{-1}[F_{m-1}(X_{m-L}^{(-n-1)}) \cdot V_i^{(-n)}] =: X_{m-1,L}^{(-n)}$$

where $X_{m,L}^{(-n-1)} = F_{m-1}[V_{m}^{(-n-1)}]$.

As one moves through the components, in decreasing order, to define the lower bounds $X_{i,L}^{(-n)}$ for $i = m-2, m-3, \ldots, 1$, there are two items of note. The first is that, the lower bound for the first component at time $-n$ depends on the lower bound for the second component at time $-n-1$ which, in turn, depends on the lower bound for the third component at time $-n-2$. Continuing in this way, we see that the lower bound on the first component at time $-n$ depends, ultimately, on the lower bound for the mth component at time $-n-m+1$, which is $X_{m,L}^{(-n-m+1)}$. Thus, in order to attempt backward coupling starting from time $-n$ to a common value at time 0, one needs to generate random vectors $V_{-n}, V_{-n-1}, \ldots, V_{-n-m+1}$ in addition to the usual $V_{-n+1}, V_{-n+2}, \ldots, V_0$. The second thing to note is that these lower bounds for our order statistics are not necessarily in order themselves. We could adjust the lower bounds as we go, for example replacing $X_{m,L}^{(-n)}$ with $\max(X_{m-1,L}^{(-n)}, X_{m,L}^{(-n)})$, or we could simply use $X_{1,L}^{(-n)}$ as a lower bound for $X_i^{(-n)}$ for all $i = 1, 2, \ldots, m$.

In this paper we chose the later approach, for simplicity, even though it will result in a small increase in the backward coupling time. $X_{1,L}^{(-n)}$ is computed as follows.
Let
\[ X_m^{(-n-m+1)} = F_m^{-1}[V_m^{(-n-m+1)}]. \] (2.9)

For \( i = m - 1, m - 2, \ldots, 2 \), compute
\[ X_i^{(-n-i+1)} := F_i^{-1}[F_i(X_{i+1}^{(-n-i)}) \cdot V_i^{(-n-i+1)}]. \] (2.10)

Set
\[ X_1^{(-n)} = F_1^{-1}[F_1(X_2^{(-n-1)}) \cdot V_1^{(-n)}]. \] (2.11)

Then \((X_1^{(-n)}, X_2^{(-n)}, \ldots, X_m^{(-n)})\) is a componentwise lower bound for any paths coming through from earlier time points.

### 2.3.2 The Algorithm

Now that we have starting values for upper and lower processes bounding each component of the sample path of interest for some time \(-n\), we evolve the paths forward to time 0 with a Gibbs sampler and hope to see \(X_i^{(0)}\) close to \(X_i^{(0)}\) for all \(i = 1, 2, \ldots, m\). We aim to ensure that
\[ \sum_{i=1}^{m} (X_i^{(0)} - X_i^{(0)})^2 < \varepsilon. \] (2.12)

If (2.12) is not satisfied, we start further back in time while making sure to reuse any previously generated \(V_i^{(-k)}\) on subsequent passes over time steps \(-k\) closer to 0. Details are given in Algorithm 2.

### 2.4 Examples

In this Section, we give several examples and simulation results for the \(\varepsilon\)-perfect Gibbs ocinid sampler which show the performance of the sampler for scale parameter distributions, shape parameter distributions, heavy-tailed distributions, and an example where the \(f_i\) in (2.1) come from completely different families.

#### 2.4.1 Exponential Distributions

Consider the component distribution densities given by
\[ f_i(x) = \theta_i e^{-\theta_i x} I_{(0,\infty)}(x), \]
Algorithm 2 Perfect Gibbs sampler for ocinid order statistics

Initialization: Set \( n = 1 \) and set an error tolerance \( \varepsilon > 0 \). Generate and store \( m \)-dimensional independent vectors \( V_{-m}, V_{-m+1}, \ldots, V_0 \) where each is populated with iid \( \text{Uniform}(0,1) \) random variates. Set \( exit = FALSE \).

while \( exit = FALSE \) do
  Compute \( X^{(-n)}_{1,U}, X^{(-n)}_{2,U}, \ldots, X^{(-n)}_{m,U} \) using (2.4), (2.5), and (2.6).
  Compute \( X^{(-n)}_{1,L} \) using (2.9), (2.10), and (2.11). Set \( X^{(-n)}_{i,L} = X^{(-n)}_{1,L} \) for \( i = 2, 3, \ldots, m \).
  for \( k = n - 1 \) down to 0 do
    Compute
    \[
    X^{(-k)}_{1,L} = F_{-1}^{-1}[F_i(X^{(-k-1)}_{2,L}) \cdot V_i^{(-k)}] \quad \text{and} \quad X^{(-k)}_{1,U} = F_{-1}^{-1}[F_i(X^{(-k-1)}_{2,U}) \cdot V_i^{(-k)}].
    \]
    Compute
    \[
    X^{(-k)}_{i,L} = F_{-1}^{-1}[F_i(X^{(-k)}_{i-1,L}) + (F_i(X^{(-k-1)}_{i-1,L}) - F_i(X^{(-k)}_{i-1,L})) \cdot V_i^{(-k)}] \quad \text{and} \quad X^{(-k)}_{i,U} = F_{-1}^{-1}[F_i(X^{(-k)}_{i-1,U}) + (F_i(X^{(-k-1)}_{i-1,U}) - F_i(X^{(-k)}_{i-1,U})) \cdot V_i^{(-k)}].
    \]
    for \( i = 2, 3, \ldots, m - 1 \).
    Compute
    \[
    X^{(-k)}_{m,L} = F_{-1}^{-1}[F_m(X^{(-k)}_{m-1,L}) + (1 - F_m(X^{(-k-1)}_{m-1,L})) \cdot V_m^{(-k)}] \quad \text{and} \quad X^{(-k)}_{m,U} = F_{-1}^{-1}[F_m(X^{(-k)}_{m-1,U}) + (1 - F_m(X^{(-k-1)}_{m-1,U})) \cdot V_m^{(-k)}].
    \]
  end
  if \( \sum_{i=1}^{m} (X^{(0)}_{i,L} - X^{(0)}_{i,U})^2 < \varepsilon \) then
    Output \( \left( \frac{X^{(0)}_{1,L} + X^{(1)}_{1,U}}{2}, \frac{X^{(0)}_{1,U} + X^{(1)}_{1,U}}{2}, \ldots, \frac{X^{(0)}_{m,L} + X^{(0)}_{m,U}}{2} \right) \).
    Set \( exit = TRUE \).
  else
    Let \( n = n + 1 \). Generate and store an \( m \)-dimensional vector \( V_{-n-m+1} \) of iid \( \text{Uniform}(0,1) \) random variates.
  end
end
We simulated 100,000 ocinid order statistics with \( m = 4 \), \( \theta = (8, 6, 4, 2) \) and \( \varepsilon = 0.0001 \). Histograms of the results are shown in Figure 2.2. The superimposed curves are the exact marginal pdfs which we were able to compute for this example. The mean BCT was 7.4 with a minimum of 1 and a maximum of 17. In Table 2.1 we show the mean backward coupling times for various \( m \) and inverse scale parameters \( \theta_i \) chosen to be smaller, larger, and with more range. For this example, the \( \theta_i \) are chosen in decreasing order, though this is not a requirement for our algorithm. Using increasing or other orderings for \( \theta \) will increase the backward coupling time, though there is usually not a dramatic difference. For example, using \( \theta = (2, 4, 6, 8) \) instead of \( \theta = (8, 6, 4, 2) \) increased the mean BCT in 100,000 draws from 7.4 to 11.2.

Table 2.1: Mean backward coupling times for the exponential, Weibull, Cauchy, and Pareto distributions

<table>
<thead>
<tr>
<th>( m )</th>
<th>( \theta )</th>
<th>exponential</th>
<th>Weibull</th>
<th>Cauchy</th>
<th>Pareto</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>(8, 6, 4, 1)</td>
<td>8.4</td>
<td>5.6</td>
<td>12.1</td>
<td>9.1</td>
</tr>
<tr>
<td></td>
<td>(20, 5, 2, 1)</td>
<td>6.9</td>
<td>4.8</td>
<td>11.9</td>
<td>7.8</td>
</tr>
<tr>
<td></td>
<td>(1.2, 0.8, 0.2, 0.05)</td>
<td>8.1</td>
<td>5.6</td>
<td>12.2</td>
<td>20.7</td>
</tr>
<tr>
<td></td>
<td>(50, 30, 20, 10)</td>
<td>6.2</td>
<td>4.3</td>
<td>11.0</td>
<td>6.2</td>
</tr>
<tr>
<td>8</td>
<td>(20, 14, 10, 8, 6, 5, 4, 2)</td>
<td>24.1</td>
<td>11.3</td>
<td>45.8</td>
<td>26.5</td>
</tr>
<tr>
<td></td>
<td>(50, 12, 10, 6, 4, 2, 0.5, 0.1)</td>
<td>16.0</td>
<td>9.0</td>
<td>32.8</td>
<td>38.6</td>
</tr>
<tr>
<td></td>
<td>(5, 2, 1.9, 1.2, 0.6, 0.4, 0.2, 0.1)</td>
<td>23.3</td>
<td>10.7</td>
<td>42.0</td>
<td>47.2</td>
</tr>
<tr>
<td></td>
<td>(100, 70, 50, 30, 20, 10, 5, 1)</td>
<td>13.4</td>
<td>7.7</td>
<td>26.9</td>
<td>14.2</td>
</tr>
<tr>
<td>12</td>
<td>(20, 18, 14, 12, 10, 8, 7, 6, 5, 4, 2, 1)</td>
<td>46.0</td>
<td>19.0</td>
<td>87.4</td>
<td>78.7</td>
</tr>
<tr>
<td></td>
<td>(70, 50, 14, 12, 10, 8, 7, 6, 5, 4, 0.2, 0.01)</td>
<td>36.6</td>
<td>16.4</td>
<td>72.3</td>
<td>80.1</td>
</tr>
<tr>
<td></td>
<td>(4.5, 3.5, 3.2, 2.19, 1.2, 0.6, 0.4, 0.3, 0.2, 0.1)</td>
<td>44.3</td>
<td>18.7</td>
<td>80.5</td>
<td>91.7</td>
</tr>
<tr>
<td></td>
<td>(100, 90, 80, 70, 50, 40, 30, 20, 10, 8, 5, 1)</td>
<td>26.2</td>
<td>12.2</td>
<td>59.1</td>
<td>32.3</td>
</tr>
</tbody>
</table>

2.4.2 Weibull Distributions

Consider now the component distribution densities given by

\[
f_i(x) = \alpha \theta_i^\alpha x^{\alpha-1} e^{-\theta_i x^\alpha} I_{(0,\infty)}(x)
\]

with \( \alpha = 3 \). We simulated 100,000 ocinid order statistics with \( m = 4 \), \( \theta = (8, 6, 4, 2) \) and \( \varepsilon = 0.0001 \). Histograms of the results are shown in Figure 2.3. Again, we were able to compute the exact marginal pdfs which are superimposed. The mean BCT was 4.6 with a minimum of 1 and a maximum of 9. Other backward coupling times for various \( m \) and parameter choices are included in Table 2.1.
Figure 2.2: Simulation results for the exponential distribution with $m = 4$, $\theta = (8, 6, 4, 2)$ and $\varepsilon = 0.0001$ with exact marginal densities superimposed. Light grey curves are densities for the remaining components for reference.
Figure 2.3: Simulation results for the Weibull distribution with $m = 4$, $\alpha = 3$, $\theta = (8, 6, 4, 2)$ and $\varepsilon = 0.0001$ with exact marginal densities superimposed. Light grey curves are densities for the remaining components for reference.
2.4.3 Cauchy Distributions

As a heavy-tailed example where computation of the exact marginal pdfs is intractable, we simulated 100,000 ocinid order statistics using the Cauchy pdfs

\[ f_i(x) = \frac{\theta_i}{\pi} \frac{1}{1 + (\theta_i x)^2} \]

to define \( f_i(x) = \theta_i f_0(\theta_i x) \). We again used \( m = 4, \theta = (8, 6, 4, 2) \) and \( \varepsilon = 0.0001 \). Histograms of the results are shown in Figure 2.4. The mean BCT was 11.1 with a minimum of 4 and a maximum of 37. Other backward coupling times are given in Table 2.1.

2.4.4 Pareto Distributions

As a heavy-tailed example with a shape, as opposed to scale, parameter, we simulated 100,000 ocinid order statistics using the Pareto pdfs

\[ f_i(x) = \frac{\theta_i}{(1 + x)^{\theta_i + 1}} I_{(0,\infty)}(x) \]

We again used \( m = 4, \theta = (8, 6, 4, 2) \) and \( \varepsilon = 0.0001 \). Histograms are shown in Figure 2.5. The mean BCT in 100,000 draws was 9.1 with a minimum of 2 and a maximum of 22. Other backward coupling times are given in Table 2.1. We can see expected spikes in backward coupling times in the cases where some \( \theta_i \leq 1 \) and the mean of the corresponding marginal distribution is infinite.

2.4.5 Exponential, Weibull, and a “Folded” Cauchy Distribution

As an example where the \( f_i \) come from different families of distributions, we simulated 100,000 values from (2.1) using

\[ f_1(x) = 2e^{-2x} I_{(0,\infty)}(x), \]
\[ f_2(x) = 3(2x)^2 e^{-(2x)^3} I_{(0,\infty)}(x), \]
\[ f_3(x) = \frac{2}{\pi} \frac{1}{1 + (2x)^2} I_{(0,\infty)}(x). \] (2.13)

The average BCT in 100,000 draws was 5.2 (with \( \varepsilon = 0.0001 \)) with a minimum of 1 and a maximum of 23. The resulting marginal distributions for \( X_1, X_2, \) and \( X_3 \) are shown in Figure 2.6.
Figure 2.4: Simulation results for the Cauchy distribution with \( m = 4, \theta = (8, 6, 4, 2) \) and \( \varepsilon = 0.0001 \)
Figure 2.5: Simulation results for the Pareto distribution with $m = 4, \theta = (8, 6, 4, 2)$ and $\varepsilon = 0.0001$. 
Figure 2.6: Simulation results with $m = 3$, different base distributions given by (2.13), and $\varepsilon = 0.0001$
2.4.6 Bayesian Principal Components Analysis

Principal component analysis (PCA) [30] is a widely used method for transforming a data set in order to convert a collection of possibly correlated variables into a collection of linearly uncorrelated variables. The uncorrelated variables, known as principal components, form a basis for reconstructing the data set. The principal components can be ranked according to their importance in explaining variability in the data and often lower ranked components can be dropped, creating a smaller basis that can recreate the original data set with varying degrees of accuracy. In 1997, Tipping and Bishop [60] introduced a density-based variant of PCA, known as probabilistic principal component analysis (PPCA) which allows for many extensions of the analysis technique to situations that are better suited for data generated from probabilistic models. In 1999 Bishop [7] first used the probabilistic framework of PPCA in order to give a fully Bayesian treatment to estimation of model parameters. Most importantly, Bayesian principal component analysis (BPCA) allows one to automatically choose the effective dimensionality by maximizing a marginal likelihood. As with most Bayesian estimation and model selection problems, one is required to employ Markov chain Monte Carlo (MCMC) methods in order to sample from posterior distributions for the purpose of inference.

We wish to relate a $d$-dimensional observed vector $\mathbf{y}$ to a $q$-dimensional vector $\mathbf{x}$ of latent variables with $1 \leq q < d$. The most common model is the linear model given by

$$
\mathbf{y} = W\mathbf{x} + \mu + \mathbf{\epsilon}
$$

where $W$ is a $d \times q$ matrix relating $\mathbf{x}$ to $\mathbf{y}$, $\mu$ is a $d \times 1$ vector of constants and $\mathbf{\epsilon}$ is a $d \times 1$ vector of random variables. It is typically assumed (e.g. [60]) that $\mathbf{x} \sim N(\mathbf{0}, I_q)$ where $I_q$ is the $q \times q$ identity matrix and that $\mathbf{\epsilon} \sim N(\mathbf{0}, \sigma^2 I_d)$ is isotropic noise. We then have

$$
\mathbf{y} | \mathbf{x} \sim N(W\mathbf{x} + \mu, \sigma^2 I_d)
$$

and it is routine to show then that

$$
\mathbf{y} \sim N(\mu, V)
$$

where $V = WW^T + \sigma^2 I_d$. 
Given $N$ observations $\vec{y}_1, \vec{y}_2, \ldots, \vec{y}_N$ from this model, the joint pdf is

$$f(\vec{y}_1, \vec{y}_2, \ldots, \vec{y}_N | \vec{\mu}, W, \sigma^2) = (2\pi)^{-dN/2} |V|^{-N/2} \exp \left[ -\frac{1}{2} \sum_{n=1}^{N} (\vec{y}_n - \vec{\mu})^T V^{-1} (\vec{y}_n - \vec{\mu}) \right]$$

which leads to a likelihood given by

$$L_q(\vec{\mu}, W, \sigma^2) = |V|^{-N/2} \exp \left[ -\frac{N}{2} tr(V^{-1} S) \right]$$

(2.14)

where $S = \frac{1}{N} \sum_{n=1}^{N} (\vec{y}_n - \vec{\mu})(\vec{y}_n - \vec{\mu})^T$.

Following Minka [44], we reparameterize the model by expressing $W$ as $W = U_{d,q}(\Lambda_q - \sigma^2 I_q)^{1/2}$ and extending the $d \times q$ matrix $U_{d,q}$ of principal eigenvectors of $S$ to a $d \times d$ matrix $U = (U_{d,q}, U_{d-q})$ with an orthogonal $(d-q) \times (d-q)$ matrix $U_{d-q}$ so that $U$ is the matrix whose columns are all eigenvectors of $S$ ordered by corresponding descending eigenvalues.

Define $\Lambda$ to be the $d \times d$ matrix

$$\Lambda = \begin{bmatrix} \Lambda_q - \sigma^2 I_q & 0 \\ 0 & 0 \end{bmatrix}.$$ 

Then we have $U \Lambda U^T = W W^T$ and

$$V = W W^T = \sigma^2 I_d = U \begin{bmatrix} \Lambda_q & 0 \\ 0 & \sigma^2 I_{d-q} \end{bmatrix} U^T.$$ 

Writing the eigendecomposition of $\hat{S}$ as $S = AGA^T$ where $G = diag(g_1, g_2, \ldots, g_d)$ is the diagonal matrix of descending eigenvalues of $\hat{S}$, we can rewrite the trace in the exponent of (2.14) as

$$tr(V^{-1} S) = tr(U B U^T A G A^T)$$

where

$$B = \begin{bmatrix} \Lambda_q & 0 \\ 0 & \sigma^2 I_{d-q} \end{bmatrix}^{-1}.$$ 

Note that $A$ is the MLE of $U$. Following [63], we simplify things by setting $U = A$. We then have that

$$tr(V^{-1} S) = tr(A B A^T A G A^T) = tr(A^T A B A^T A G) = tr(B G) = \sum_{i=1}^{q} \frac{g_i}{\lambda_i} + \sum_{i=q+1}^{d} \frac{g_i}{\sigma^2}$$
and, for fixed $q$, (2.14) can be rewritten as

$$L_q(\lambda_1, \lambda_2, \ldots, \lambda_q, \sigma^2) = \left( \prod_{i=1}^{q} \lambda_i \right)^{-N/2} (\sigma^2)^{-N(d-q)/2} \exp \left[ -\frac{N}{2} \sum_{i=1}^{q} \frac{g_i}{\lambda_i} \right] \cdot \exp \left[ -\frac{N}{2\sigma^2} c(q) \right]$$

(2.15)

where $c(q) := \sum_{i=q+1}^{d} g_i$.

The Priors

We will a priori take $q$ to be uniformly distributed on $\{1, 2, \ldots, d-1\}$, though other choices of priors can easily be used with our algorithm. It is common in the literature to require that $\lambda_1 > \lambda_2 > \cdots > \lambda_q > \sigma^2$ for identifiability and for $(\lambda_1, \lambda_2, \ldots, \lambda_q, \sigma^2)$ to have the distribution of the order statistics for $q+1$ iid inverse gamma ($IG$) random variables with hyperparameters $\alpha$ and $\beta$ which we will assume are fixed. To be clear, we are using $\beta$ as an inverse scale parameter. Specifically, if $X \sim IG(\alpha, \beta)$ the pdf is

$$IG(x; \alpha, \beta) = \frac{1}{\Gamma(\alpha)} \beta^\alpha x^{-(\alpha+1)} e^{-\beta/x} I_{(0,\infty)}(x).$$

Define $\theta_q = (\lambda_1, \lambda_2, \ldots, \lambda_q, \sigma^2)$. The prior on $q$ and $\theta_q$ is

$$\pi(q, \theta_q) = \pi(\theta_q|q) \pi(q)$$

$$= (q + 1)! \cdot \prod_{i=1}^{q} IG(\lambda_i; \alpha, \beta) \cdot IG(\sigma^2; \alpha, \beta) \cdot I_{(\lambda_1 > \lambda_2 > \cdots > \lambda_q > \sigma^2)}$$

$$
\cdot \frac{1}{d-1} I_{(1,2,\ldots,d-1)}(q).
$$

The Posterior “Target” Let $D := \{\bar{y}_1, \bar{y}_2, \ldots, \bar{y}_n\}$. The posterior distribution is then

$$\pi(q, \theta_q|D) \propto L_q(\lambda_1, \lambda_2, \ldots, \lambda_q, \sigma^2) \cdot \pi(\theta_q|q) \pi(q)$$

(2.16)

This is the target distribution that we wish to sample from. Note that $\lambda_1, \lambda_2, \ldots, \lambda_q, \sigma^2$ are OCINID order statistics. In particular, we have the marginal distributions

$$\lambda_i \sim IG \left( \frac{N}{2} + \alpha, \frac{N g_i}{2} + \beta \right),$$

for $i = 1, 2, \ldots, q$, and

$$\sigma^2 \sim IG \left( \frac{N(d-q)}{2} + \alpha, \frac{N c(q)}{2} + \beta \right).$$
along with the restrictions $\lambda_1 > \lambda_2 > \cdots, \lambda_q > \sigma^2$.

In [63], the authors sample from (2.16) using a $(q+1)$-dimensional Gibbs sampler within a reversible jump MCMC algorithm to move $q$ through the set \{1, 2, $\ldots$, $d-1$\}. The $q+1$ parameters each have individual, and non-perfect, Gibbs moves. While we could use our perfect OCINID Gibbs algorithm to sample the parameters in one go between each birth and death move for $q$, we have instead chosen to forgo the reversible jump aspect and, instead, to repeatedly sample $\theta_q$ for each fixed value of $q$ in order to estimate a maximum a posteriori (MAP) estimator which can be used for model selection. In particular, we will compare the evidence for each subspace dimensionality, using Laplace’s method and the Bayesian Information Criterion (BIC) as in [34] and [44].

We generated a data set of 100 points from an 8-dimensional Gaussian distribution with variances 10, 8, 6, 4, 2, 0.5, 0.5, and 0.5. The observed sample variances were 9.5682, 7.8954, 5.8364, 3.8566, 2.0067, 0.5419, 0.5222, 0.4994, 0.4572, and 0.4425, respectively. After sampling $n = 10,000$ values of $\theta_q$ for each of $q = 1, 2, \ldots, 9$, we used the resulting MAP estimates to compute the maximum log-likelihood for each $q$, the BIC evidence ( [34], [44]) given by
\[
\pi(y|q) \approx \left( \frac{\pi_{\lambda}}{\pi_{\sigma}} \right)^{-n/2} \frac{1}{n^{-(k+q/2)}} (2.17)
\]
where $k = dq - q(q+1/2)$, and the Laplace evidence ( [34], [44]) given by
\[
\pi(y|q) \approx 2^{(k-q)/2} \pi_{\sigma}^{n(d-q)} n^{-q/2} |A|^{-1/2} \prod_{i=1}^{q} \Gamma \left( \frac{d-i}{2} \right) (2.18)
\]
for
\[|A| = n^k \prod_{i=1}^{q} \prod_{j=i+1}^{d} (\lambda^{-1}_j - \lambda^{-1}_i)(\lambda_i - \lambda_j)\]
with
\[\lambda_i = \begin{cases} 
\lambda_i, & \text{if } i \leq q \\
\frac{1}{d-q} \sum_{j=q+1}^{d} \hat{\lambda}_i, & \text{if } q < i \leq d.
\end{cases}\]
The idea would be to select models that maximize (2.17) and/or (2.18). We refer the interested reader to [34] and [44] for more information about these criteria.

The results are given in Figure 2.7 where, as expected, the maximized log-likelihood is increasing in $q$. The BIC appears to choose $q = 5$ principal components, which makes sense given the variances
(10, 8, 6, 4, 2, 0.5, 0.5, 0.5) used to simulate the data. The Laplace evidence also chooses $q = 5$, though it is not as pronounced as the BIC result. The average BCT in 10,000 draws for this 8-dimensional model, with $\alpha = 2$ and $\beta = 3$, was 38 time steps.

![Graphs showing scoring the models](image)

**Figure 2.7**: Scoring the models

### 2.5 Conclusion

The perfect OCIND Gibbs algorithm appears to be performing quite well for all of the examples we have tried, including the non-toy problem of order selection in Bayesian principal components analysis. Although it makes intuitive sense that $X^{(-n)}_i$ getting close to $X^{(-n)}_i$ for some $i$ will tend to make $X^{(-n)}_j$ and $X^{(-n)}_j$ close for $j = i - 1, i + 1$, the upper and lower paths for each component may also move apart again in the next time step. There always appears to be a point in time where all $m$ upper and lower processes achieve an $\varepsilon$-coupling, at which point they will stay coupled, and it seems that this happens much faster than would be expected based, say, on good fortune that the uniform random variates $V^{(-n)}_i$ from Section 2.3 all happen to line up in the right way. We see changes in average backward coupling times consistent with different choices of parameters and distributions with heavier or lighter tails. For example, in Section 2.4.1, where the distributions are all exponential, if the rate parameters are decreasing so that the minimum OCINID order statistic has the highest rate and the maximum has the lowest rate, it is not surprising that coupling is quite fast. However, for different orderings of parameters, coupling is slower, as expected, but not significantly so. It is our hope for the future to be able to prove that coupling will happen in finite time.
and to be able to say something about the coupling rate, as least for some classes of distributions, if not in
general.
Chapter 3

Cross-Pollination

This chapter presents a paper I wrote with Dr. Jem Corcoran, Dr. Michael Schneider, and Dr. Jason Bernstein, which can be found at https://arxiv.org/abs/2010.13921.

This work has been presented at seminars at Lawrence Livermore National Lab.

3.1 Contribution

We present a Bayesian data fusion method to approximate a posterior distribution from an ensemble of particle estimates that only have access to subsets of the data. We advance the work Schneider et al. [56] by showing the standard importance weighting scheme, as opposed to the deterministic weighting scheme, to fuse estimates is more effective when constrained to our set-up. Our approach relies on approximate probabilistic inference of model parameters through Monte Carlo methods, followed by an update and resample scheme related to multiple importance sampling to combine information from the initial estimates. Furthermore we show our method is convergent in the particle limit. We demonstrate that fusing these estimates with versions of Gaussian mixture models or consensus Monte Carlo methods are biased as “fractioning” the prior or “tempering” the likelihood are not immediately applicable. We show our method is directly suited to application on multi-sensor data fusion problems by demonstrating efficacy on a multi-sensor Keplerian orbit determination problem and a bearings-only tracking problem.
3.2 Introduction

Data fusion is the process through which different sources of information are combined to form a joint estimate of a process, target, or distribution of interest. There are many methods for data fusion, especially in the areas of multi-sensor measurements and target tracking [4,41]. Several data fusion techniques exist to combine analytic descriptions or numerical estimates of probability density functions (pdfs) [24]. Optimal Bayesian fusion of pdfs [14] follows Bayes’ rule for conditional density inference in multiplying individual pdfs followed by division by the joint marginal distribution of the data. The normalised weighted geometric mean rule [2], an extension to non-Gaussian pdfs of the covariance intersection rule [31], avoids division by the marginal data density by direct computation of numerical weights on each function involved in the product of individual measurement pdfs.

To be applicable to non-linear and non-Gaussian problems, many methods must approximate distributions through sampling techniques. Sampling methods complicate the aforementioned fusion strategies as additional steps of estimation are then required to make pdf multiplication feasible. If the sampling-based approximation of the pdfs are Dirac mixture measures, multiplication must be applied after a process such as kernel density estimation to return non-zero estimates of the pdf products [42]. We present cross-pollination, an alternative Bayesian sampling-based strategy, that uses multiple importance sampling ideas. Schneider et al. [56] adapted the multiple importance sampling work of Elvira et al. [19], to a sensor data fusion task through a multiple importance sampling framework. We improve this method via guarantees of convergence in the particle limit and validation of consistency of different weighting schemes, thus providing a data fusion strategy to accurately quantify uncertainty while avoiding the need for multiplication of pdf approximations by working directly with the existing particle estimates.

3.3 Method

3.3.1 Objective, Set-Up, and Requirements

The goal of cross-pollination is to approximate a posterior distribution $P(\theta | D)$, conditioned on a set of data or observations $D$, from an ensemble of existing particle estimates of partial posterior distributions
\( P(\theta \mid D_j) \), where the \( D_j \ (j = 1, \ldots, M) \) form a partition of \( D \).

The random variable of interest \( \theta \) can take various forms e.g., an unknown value as in Section 3.5.1, a set of parameters which drives a deterministic motion model as in Section 3.5.3, or a time-indexed collection of states which estimate a full trajectory as in Section 3.5.4. The initial particle estimates can be obtained online through importance sampling-resampling, or be pre-processed samples that contain the information from \( D_j \) through other Monte Carlo or Markov chain Monte Carlo techniques. Cross-pollination can be applied in batches or sequentially, through repeated applications of the main steps of our weighting scheme and resampling.

There are a few requirements to apply cross-pollination. The first requirement is that a common prior is used across all the estimates of the random variable of interest. The second is that likelihood functions must be available for all observations. These requirements are essential for Bayesian calculations and are included here for completeness. The third requirement is that the measurements must be independent, this is a common assumption in sampling-resampling frameworks as it allows for joint likelihood functions to be expressed as products of individual likelihood functions [58]. The fourth requirement, which we will refer to as the core requirement, is that particles obtained from any of the estimates must be able to map into the domain of the likelihood functions of the complement observations. This requirement is what allows our weighting scheme to occur. We will see that in the inference of parameters in a deterministic motion model Section 3.5.3 the core requirement is easily satisfied, while in the case of a stochastic motion model more work is required to satisfy the core requirement Section 3.5.4.

3.3.2 Measure Theoretic Description of Method

In the discussion of the method that follows we will assert the data \( D = (z_1, z_2, \ldots, z_M) \) was partitioned into single observations, \( D_j = z_j \), and that the initial samples were obtained through importance sampling. Both of these choices are only here for the purpose of clarity; we will divert from this set up in the applications in Section 3.5. To speak accurately about updating the samples we turn to a measure theoretic viewpoint.

Let \( \pi \) be the probability measure corresponding to the posterior distribution \( P(\theta \mid D) \), \( \pi_{j,0} \) be the probability measure corresponding to the initial partial posterior distribution \( P(\theta \mid D_j) \). We denote the
probability measures that estimate these measures as $\pi^N$ and $\pi^N_{j,0}$ respectively, noting that the superscript $N$ indicates that these estimates are each comprised of $N$ particles approximately sampled from their respective distributions.

Let $\theta$ be a random variable of interest with prior $P(\theta)$ and observations $D = (z_1, z_2, \ldots, z_M)$ with measurement uncertainty which is encapsulated by a likelihood function. For each member of the partitioned data $D_j$, a Dirac mixture measure $\pi^N_{j,0}$ is formed from importance sampling using a likelihood function associated with the observation, $g_j(\theta)$, where $P(D_j \mid \theta) \propto g_j(\theta)$. Specifically,

$$\pi^N_{j,0} = \frac{1}{N} \sum_{i=1}^{N} \delta_{\theta(i)}(j = 1, \ldots, M), \quad (3.1)$$

where the particles $\theta(i)(i = 1, \ldots, N)$ are obtained through importance sampling with the common prior $P(\theta)$ as the importance distribution and the likelihood function of the observation used to weight and resample. We denote the Dirac function $\delta(x - x^*)$ as $\delta_{x^*}$. We are to weight the particles of our initial measure $\pi^N_{j,0}$, which is built from a single observation, to form an empirical measure $\pi^N_j$ that estimates $\pi$.

We use the Radon-Nikodym derivative to obtain the weights needed to match expectations between the two measures. As $\pi$ is absolutely continuous with respect to $\pi_{j,0}$ we have that

$$\frac{d\pi}{d\pi_{j,0}}(\theta) = \frac{P(D_{-j} \mid \theta) \prod_{k \neq j} g_k(\theta)}{P(D_{-j} \mid D_j) P(D_j \mid \theta) P(\theta \mid D_j)}, \quad (3.2)$$

where $D_{-j} = (z_1, z_2, \ldots, z_{j-1}, z_{j+1}, \ldots, z_M)$ and the final product is due to the independence of the measurements. Each particle is now updated with a weight that corresponds to the value in the product of likelihood functions of the complementary data. We refer to this step as cross-epoch weighting.

$$\pi^N_j = \sum_{i=1}^{N} w_j(i) \delta_{\theta(i)}, \quad w_j(i) = \frac{\tilde{w}_j(i)}{\sum_{i'=1}^{N} \tilde{w}_j(i')} , \quad \tilde{w}_j(i) = \prod_{k \neq j} g_k(\theta(i)). \quad (3.3)$$

Notice that the structure of the weights is reflected in an application of Bayes’ rule working directly with the probability distribution functions,

$$P(\theta \mid D) = P(\theta \mid D_{-j}, D_j) = \frac{P(D_{-j} \mid \theta, D_j)}{P(D_{-j} \mid D_j)} P(\theta \mid D_j) = \frac{P(D_{-j} \mid \theta)}{P(D_{-j} \mid D_j)} P(\theta \mid D_j). \quad (3.4)$$

Now, we combine the empirical measures through pooling and resampling to obtain $N$ equally weighted particles. One way to achieve this is through combining all of the measures with a summation and dividing
by the total number of measures

$$\pi^{MN} = \frac{1}{M} \sum_{j=1}^{M} \pi_j = \frac{1}{M} \sum_{j=1}^{M} \sum_{i=1}^{N} w_j^{(i)} \delta_{\theta_j^{(i)}},$$

(3.5)

and then employing a resampling technique, say multinomial, to obtain our final approximation

$$\pi^N = \frac{1}{N} \sum_{i=1}^{N} \delta_{\theta^{(i)}},$$

(3.6)

where $\theta^{(i)} (i = 1, \ldots, N)$ are identically and independently distributed from the measure $\pi^{MN}$.

We can fuse the particles in another way; instead of normalizing the weights by data partition, as in (3.3), we can go straight to a fused measure by normalizing all the weights together,

$$\pi_t^{MN} = \sum_{j=1}^{M} \sum_{i=1}^{N} w_j^{(i)} \delta_{\theta_j^{(i)}}, \quad w_j^{(i)} = \frac{\bar{w}_j^{(i)}}{\sum_{j'=1}^{M} \sum_{i'=1}^{N} \bar{w}_{j'}^{(i')}}, \quad \bar{w}_j^{(i)} = \prod_{k \neq j} g_k \left( \theta_k^{(i)} \right),$$

(3.7)

Then, in the same fashion we would employ a sampling technique to get to our final estimate of $N$ particles,

$$\pi_t^N = \frac{1}{N} \sum_{i=1}^{N} \delta_{\theta^{(i)}},$$

(3.8)

where $\theta^{(i)} (i = 1, \ldots, N)$ are identically and independently distributed from the measure $\pi_t^{MN}$. We refer to the methods as ‘norming-apart’ and ‘norming-together’, respectively. This norming-apart process is diagrammed in Figure 3.1 both methods are proved to converge under a root mean square distance in the particle limit when the initial estimates are formed using importance sampling. The proof of the norming-apart method is presented in Section 3.3.4. The ease of these formulations suggest straightforward algorithms for both methods, pseudo-code is provided in Section 3.7.
3.3.3 Relation to Multiple Importance Sampling

Importance sampling is a well-known technique of sampling from a known distribution $q(\theta)$ and adjusting the samples with importance weights to approximate a target distribution $\pi(\theta)$ [52]. The typical form of the importance weights for a sample $\theta^{(i)}$ is $w(\theta^{(i)}) = \pi(\theta^{(i)})/q(\theta^{(i)})$. Multiple importance sampling expands this idea by introducing a number of proposal distributions $q_j(\theta)$ ($j = 1, \ldots, M$) that can be adjusted in various ways to estimate the desired distribution. Both the weighting and ordering of selection from proposal distributions can affect the final estimate [19].
The connection to cross-pollination is formed by considering the importance sampling target distribution as the posterior conditioned on all the data \( \pi(\theta) = P(\theta | D) \) and the importance sampling proposal distributions to be the marginal posteriors \( q_j(\theta) = P(\theta | D_j) \), which makes the initial measures \( \pi_{j,0} \) the collections of samples obtained from each of our proposal distributions. By using the data-dependent proposals we have created a data fusion algorithm that can make use of existing multiple importance sampling theory. Both the norming-apart and norming-together cross-pollination schemes resemble aspects of generic diverse population Monte Carlo methods discussed by Elvira et al. [18]. Schneider’s original work on cross-pollination [56] used the deterministic mixture weights proposed by Elvira et al. [19]. In this case, one would have,
\[
\pi_{MN}^{dm} = \sum_{j=1}^{M} \sum_{i=1}^{N} \hat{w}_j^{(i)} \delta_{\theta_j^{(i)}}, \quad w_j^{(i)} = \frac{\hat{w}_j^{(i)}}{\sum_{j'=1}^{M} \sum_{i'=1}^{N} \hat{w}_{j'}^{(i')}}, \quad \hat{w}_j^{(i)} = \frac{\prod_{k=1}^{M} g_k(\theta_j^{(i)})}{\sum_{k=1}^{M} g_k(\theta_j^{(i)})},
\]
and, through sampling would obtain,
\[
\pi_{dm}^{N} = \frac{1}{N} \sum_{i=1}^{N} \delta_{\theta^{(i)}},
\]
where \( \theta^{(i)} (i = 1, \ldots, N) \) are identically and independently distributed from the measure \( \pi_{MN}^{dm} \). This weight formulation is of particular interest as Elvira et al. [19] catalog it as the scheme having the minimum variance of the multiple importance sampling schemes they examined. However, this scheme is only shown to be consistent, in the limit of the number of proposals, when the normalizing constant is known. This makes the deterministic mixture weights scheme less applicable to the problem at hand which demands a fixed number of proposal distributions depending on how the data has been partitioned and has no knowledge of the normalizing constant, which is typically estimated by the sum of the weights in importance sampling theory.

### 3.3.4 Proof of Method

We now prove that the final resampled measure from the norming-apart cross-pollination process, \( \pi_{MN}^{N} \), converges to \( \pi \) in the particle limit. In particular, we prove this convergence if the initial particles were sourced via importance sampling. We make use of the following “root mean square” distance, notation, and
operators on the set of probability measures presented by Law et al. [36, p.87-93]:

\[ d(\mu, \nu) = \sup_{\|f\|_\infty \leq 1} \sqrt{\mathbb{E}[\mu(f) - \nu(f)]^2}, \]  

(3.11)

where

\[ \mu(f) = \int_{\mathbb{R}^n} f(v) \mu(dv), \]  

(3.12)

\[ (L_j \mu)(d\nu) = \frac{g_j(\nu) \mu(d\nu)}{\int_{\mathbb{R}^n} g_j(\nu) \mu(d\nu)}, \]  

(3.13)

and

\[ (S^N \mu)(d\nu) = \frac{1}{N} \sum_{n=1}^{N} \delta_{\nu^{(n)}}(d\nu), \quad \nu^{(n)} \sim \mu \text{ i.i.d.}. \]  

(3.14)

Here, \( g_j \) is the likelihood function corresponding to the \( j \)th data point \( D_j \). We also make use of two lemmas proved by Law et al. [36, p.87-93]:

**Lemma 1.** The sampling operator satisfies

\[ \sup_{\mu \in \mathcal{P}(\mathbb{R}^N)} d(S^N \mu, \mu) < \frac{1}{\sqrt{N}}. \]  

(3.15)

**Lemma 2.** If there exists \( \kappa \in (0, 1] \) such that for all \( \nu \in \mathbb{R}^N \) and \( j \in \mathbb{N} \), \( \kappa \leq g_j(\nu) \leq \kappa^{-1} \), then

\[ d(L_j \nu, L_j \mu) \leq 2\kappa^{-2}d(\nu, \mu). \]  

(3.16)

We can now frame the norming-apart process as a sequence of applications of \( S^N \) and \( L_j \). We will use \( L_{-j} \) to be transforming a measure, which has incorporated the \( j \)th data, using the Radon-Nikodym derivative to incorporate the remaining data (via a product of likelihood functions) to an appropriate approximate measure of the full posterior. We can update the empirical measure equations, (3.1)-(3.6), using this notation:

\[ \pi_{j,0}^N = S^N L_j S^N P, \]  

(3.17)

\[ \pi_j^N = L_{-j} \pi_{j,0}^N = L_{-j} S^N L_j S^N P, \]  

(3.18)

\[ \pi_{MN}^M = \frac{1}{M} \sum_{j=1}^{M} \pi_j^N = \frac{1}{M} \sum_{j=1}^{M} L_{-j} S^N L_j S^N P, \]  

(3.19)

\[ \pi^N = S^N \pi_{MN}^M, \]  

(3.20)
where \( P \) is the prior distribution of \( \theta \).

Armed with this theory we can prove our result with some applications of the triangle inequality. We are loose with our application of the likelihood lemma Equation 3.16 in our proof summarizing the result as

\[
d(L_j\nu, L_j\mu) \leq C_j d(\nu, \mu).
\]

**Theorem 3.** Assumptions: Those of Equation 2 and independent measurements. Then,

\[
d(\pi^N, \pi) < \frac{C_M}{\sqrt{N}}
\]  

(3.21)

**Proof.**

\[
d(\pi^N, \pi) \leq d(\pi^N, \pi^{MN}) + d(\pi^{MN}, \pi)
\]

\[
= d(S^N \pi^{MN}, \pi^{MN}) + d(\pi^{MN}, \pi)
\]

\[
\leq \frac{1}{\sqrt{N}} + d\left( \frac{1}{M} \sum_{j=1}^{M} \pi_j^N, \pi \right)
\]

\[
\leq \frac{1}{\sqrt{N}} + \frac{1}{\sqrt{M}} \sum_{j=1}^{M} d(\pi_j^N, \pi)
\]

\[
\leq \frac{1}{\sqrt{N}} + \frac{1}{\sqrt{M}} \sum_{j=1}^{M} d(L_j \pi_j^N, L_j L_j P)
\]

\[
\leq \frac{1}{\sqrt{N}} + \frac{1}{\sqrt{M}} \sum_{j=1}^{M} C_j d(\pi_j^N, L_j P)
\]

\[
\leq \frac{1}{\sqrt{N}} + \frac{1}{\sqrt{M}} \sum_{j=1}^{M} C_j d(S^N L_j S^N P, L_j P)
\]

\[
= \frac{1}{\sqrt{N}} + \frac{1}{\sqrt{M}} \sum_{j=1}^{M} C_j d(S^N L_j S^N P, L_j P)
\]

\[
\leq \frac{1}{\sqrt{N}} + \frac{1}{\sqrt{M}} \sum_{j=1}^{M} C_j d(S^N L_j S^N P, L_j S^N P) + d(L_j S^N P, L_j P))
\]

\[
\leq \frac{1}{\sqrt{N}} + \frac{1}{\sqrt{M}} \sum_{j=1}^{M} C_j \left( \frac{1}{\sqrt{N}} + C_j d(S^N P, P) \right)
\]

\[
\leq \frac{1}{\sqrt{N}} + \frac{1}{\sqrt{M}} \sum_{j=1}^{M} C_j \left( \frac{1}{\sqrt{N}} + C_j \frac{1}{\sqrt{N}} \right)
\]

\[
= 1 + \frac{1}{\sqrt{M}} \sum_{j=1}^{M} (C_j + C_j C_j)
\]

\[
\leq \frac{1}{\sqrt{N}}
\]
We have shown convergence of the norming-apart method in the particle limit. The fourth line of the proof is not trivial, it was achieved by bounding cross terms of the squared sum by a sum of the two terms squared, that is, \( ab \leq \frac{a^2 + b^2}{2} \) followed by simple bounding arguments. By noting that the norming-together method can be written to look like the norming-apart method,

\[
\pi^N_t = \sum_{j=1}^{M} K_j \pi^N_j, \quad K_j = \frac{\sum_{i=1}^{N} u^{(i)}_j}{\sum_{j'=1}^{M} \sum_{i=1}^{N} u^{(i)}_{j'}}.
\] (3.22)

we may employ a similar proof to show that the norming-together method converges in the particle limit.

3.4 Other Methods

We discuss four other methods that can be applied or loosely applied to the proposed data fusion problem. The first two methods are fusing samples via Gaussian mixture models. We can express the posterior as a weighted multiplication of the partial posteriors that is, for three pieces of data,

\[
P(x|y_1, y_2, y_3) \approx \alpha P(x|y_1)^{w_1} P(x|y_2)^{w_2} P(x|y_3)^{w_3} \times P(y_1|x)^{w_1} P(y_2|x)^{w_2} P(y_3|x)^{w_3} P(x)^{w_1+w_2+w_3}.
\] (3.23)

So to remain as consistent as possible we require the sum of the weights \( w_i \) to be one. Now, as this process requires multiplication, the first step would be to model each of the samples of the partial posteriors as a Gaussian mixture. Then multiplying the Gaussian mixtures together could yield a probability distribution function that would approximate the full posterior. We consider a naive combination of the Gaussian mixture models which is to choose equal weights. A more refined approach would be to pick the weights as to minimize the Kullback-Leibler divergence between the fused probability density function and each of the partial posteriors as described in by McCabe et al [42].

In addition to these methods we can consider an adaptation of the consensus Monte Carlo algorithm (CMC) [57]. Typically these methods are employed for big data problems. The methods can be described in three steps: first the set of data is divided in \( M \) shards, then samples are obtained by running Monte Carlo algorithms to obtain samples from the partial posteriors with a fractioned prior, then the samples are combined across the shards using empirical covariances as weights. Thus the authors are trying to exploit
Bayes Rule in the form

\[ P(x|y) \propto \prod_{i=1}^{M} P(y_i|x)P(x)^{1/M} \]  

(3.24)

In our case, the samples are derived from the full partial posteriors (not a fractioned prior) thus the method is not immediately applicable, regardless we compare below. Liu [37] introduces the importance weighted consensus Monte Carlo algorithm which argues that to achieve better matching the samples that come out of the consensus Monte Carlo algorithm can be improved if they are given an importance weight.

### 3.5 Examples

We demonstrate the algorithms on three examples: the first is an example with an analytic posterior distribution that serves as an introduction to application of the method as well as a verification of the convergence rate, the second is an orbit determination problem where particle estimates borne of data from two different sensors are fused via the norming-together method, the third example is a bearings-only tracking example that shows one way the ideas of cross-pollination may be carried into a sequential problem with a stochastic motion model. For simplicity in the latter two examples, we consider normal or multivariate normal likelihood functions for measurements. We denote the pdf of a normal distribution with mean \( \mu \) and variance \( \sigma^2 \) evaluated at \( x \) as \( N(x; \mu, \sigma^2) \). Similarly, for a multivariate normal we will have \( MVN(x; \mu, \Sigma) \) to be the pdf of a multivariate normal with mean \( \mu \) and covariance matrix \( \Sigma \) evaluated at \( x \).

#### 3.5.1 Convergence Verification and Comparison on Gamma Example

Our theory argues for a convergence rate of \( \sqrt{N} \) of our estimator \( \pi^N \) to the probability measure \( \pi \) that corresponds to the posterior distribution. We verify this result on an example with gamma distributions. We denote that \( x \) is distributed from a gamma distribution with shape parameter \( k_0 \) and rate parameter \( \theta \) by \( x \sim \Gamma(k_0, \theta) \). First we give our random variable of interest a gamma prior distribution \( x \sim \Gamma(k_0, \theta_0) \).

Next we have three observations each with a gamma distribution so that \( y_j|x \sim \Gamma(k_j, x^{-1}) \) (\( j = 1, 2, 3 \)). This set-up admits an analytic posterior distribution,

\[ x|y_1, y_2, y_3 \sim \Gamma\left(k_0 + k_1 + k_2 + k_3, (\theta_0^{-1} + y_1 + y_2 + y_3)^{-1}\right). \]  

(3.25)
For this experiment we chose $k_0 = 5/2$, $\theta_0 = 1/2$, $y_1 = 1$, $y_2 = 2$, $y_3 = 3$, $k_1 = 4$, $k_2 = 10$, $k_3 = 25$. The sets of initial particles corresponding to $\pi^N_{j,0}$ $(j = 1, 2, 3)$ were obtained by one round of standard importance sampling. The particles were then cross-epoch weighted and resampled under both methods of norming-together and norming-apart cross-pollination. We conducted 1000 Monte Carlo trials for $N = 10^2, 10^3, 10^4, 10^5, 10^6$ particles.

In Figure 3.3 we see how the methods stack up against one another. The matching of the posterior distribution is most evident using the cross-pollination algorithm and the other methods are biased, most likely due to assumptions of Gaussianity and the fact that the initial samples come from estimates of partial posteriors—not fractioned priors or tempered likelihoods. The methods involving Gaussian mixture models only give probability distribution functions as their outputs, as sampling from these distributions is a non-trivial task [46], [29].
Figure 3.2: (a) Blue, purple, and cyan histograms show the initial sets of particles for a trial with $10^6$ particles taken created from observations 1, 2, and 3 respectively. The grey histogram is of particles that have been selected after a single run of cross-pollination. Solid pink line is the posterior pdf. (b) The black dotted lines follow $\frac{8}{\sqrt{N}}$ (top) and $\frac{1}{2\sqrt{N}}$ (bottom). Solid red lines from darkest to lightest plot the mean over 1000 Monte Carlo trials of the absolute error of the estimated first four moments (mean, variance, skew, kurtosis) against the true value of the moments for the norming-apart method. Similarly the green dashed lines describe the same situation for the norming-together method.
To reiterate, we formed the particles from the observations (labelled Obs 1 particles etc in Figure 3.2a) based on drawing particles from the prior and importance sampling with the single gamma likelihood. The particles were then put through the cross-pollination process of cross-epoch weighting and resampling to form the final estimate of particles (labelled Cross-pol particles in Figure 3.2a). Our theory is supported by Figure 3.2b as we see the mean absolute error decreases at a rate proportional to the square root of
the number of particles for each moment. The proportion changes depending on which moment is being estimated, as one would expect the proportionality constant is higher for the higher order moments.

We included this example to provide an introduction to the application of the method and check convergence rates where the posterior distribution had an analytic form. We believe cross-pollination can be a useful tool in multi-sensor data fusion applications as demonstrated in the following applications.

3.5.2 Multiple Sensor Bearings

A possible application of the cross pollination algorithm is to see what effects multiple sensor fusion can have in localization problems. For instance, is having an expensive and accurate sensor better or worse than having several cheaper sensors. We will show the changes in the shape of the uncertainty between a importance sampling with a highly accurate sensor compared cross-pollination between samples obtained from importance sampling from two less accurate sensors. Consider a two-dimensional Gaussian prior at the origin with the identity as the covariance, the true location of the object is (0.5, 0.5), sensors are located at (3, −3) and (−3, −2) and have $\sigma = 0.05$ for the less accurate fusion, the more accurate sensor is located at (−3, −2) and has $\sigma = 0.01$. We can see in Figure 3.4 the differences in the shape of the uncertainties. The uncertainty proposed by the fusion is more concentrated around the value of the truth. Operators could conduct further experiments with the cross-pollination algorithm to create and examine desired geometries when estimates are to be fused.
Figure 3.4: Hexbin plots of samples (purple to yellow dots) and the orientation of the observations (dot-dash and dashed lines) from the observers (grey-diamond) as well as the true location (red-diamond).
3.5.3 Orbit Determination

Space traffic management is becoming an increasingly challenging remote sensing and data fusion problem as the number of satellites and debris around the Earth continue to grow. A key aspect of space traffic management problems includes orbit determination. Multi-sensor data fusion for an orbital determination problem has been explored using the geometric mean density (GMD) fusion rule [42].

In an orbit determination scenario where ground-based optical angles-only measurements are accessible online and orbital measurements are lagged, practitioners may desire making ground-based estimates immediately and fusing the orbital estimates when they become available at a later time. We simulate this situation below for Keplerian orbits.

Keplerian orbits trace closed ellipses in space and are described in a six-dimensional state space, which is typically described by the six orbital elements: semi-major axis, eccentricity, inclination, argument of periapsis, right ascension of the ascending node, and the true anomaly. Equivalently, these orbits can be identified by a three-dimensional position and a three-dimensional velocity at a given time. We let \( \theta = (x, y, z, \dot{x}, \dot{y}, \dot{z}) \) at time \( t_0 \), \( P(\theta) \) be the prior, and consider two sets of three right ascension (RA) and declination (DEC) measurements occurring: one taken from a sensor in low Earth orbit (LEO) \( D_1 \) over three minutes and the other from a ground-based observer \( D_2 \) over a four minute period taking place six minutes after the final orbiting-sensor measurement. Sensor measurements are modeled as multivariate Gaussian with mean given by the true RADEC and standard deviations given by \( \sigma_{\text{LEO}} = 2 \) arcseconds and \( \sigma_{\text{ground}} = 20 \) arcseconds. Therefore, the likelihood function for measurements \( j \) of the LEO observer is,

\[
g_j\left(\theta^{(i)}\right) = \mathcal{MVN}\left(H_j\left(\theta^{(i)}\right); z_j, \Sigma_{\text{LEO}}\right),
\]

where the non-linear measurement function \( H_j \) maps the particles into the correct measurement space by propagating the orbit specified by \( \theta^{(i)} \) to the correct observation time \( t_j \) and retrieving the RADEC measurements that would have been obtained from \( j \)th sensor, and the covariance matrix is \( \Sigma = \sigma^2_{\text{LEO}}I_2 \). Of course, the ground based likelihoods take the same form but with \( \sigma_{\text{ground}} \). This set-up satisfies the requirements of cross-pollination.

We will apply cross-pollination to the estimates that comprise \( \pi_{1,0}^N \) and \( \pi_{2,0}^N \) which consist of 50,000
samples each, obtained with an MCMC sampler to produce initial orbit determinations based on the data provided. We fuse these samples using norming-together cross-pollination and perturb the samples to achieve our final estimate $\pi^N$. We diagram this process in Figure 3.5.

![Figure 3.5: From left to right: a random sample of 450 of the 50,000 particles estimates traced out orbits (grey) and the true orbit (red) for the LEO sensor, ground-based sensor, and the estimates after being refined through the norming-together cross-pollination process.](image)

From Figure 3.5 we can see that the refined orbits from the norming-together cross-pollination process are providing an improved picture of the situation and uncertainty. We show the improvements to root mean square error in the position and velocity of the samples in Table 3.1. We can see in Figure 3.5 a distinct improvement in the uncertainty as the ground sensor MCMC process allowed many samples from a degenerate orbital plane, due to a small observation window and large uncertainty in the measurements, that has been eliminated in the fused results. Furthermore, the distinct “banana shaped uncertainty” [28] that emerges in orbit determination problems is seen in the rightmost image of Figure 3.5, indicating uncertainty realism.
Table 3.1: RMSE on Orbit Samples

<table>
<thead>
<tr>
<th>Source of Samples</th>
<th>Position RMSE</th>
<th>Velocity RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>LEO</td>
<td>280 km</td>
<td>104 m/s</td>
</tr>
<tr>
<td>Ground</td>
<td>681 km</td>
<td>2243 m/s</td>
</tr>
<tr>
<td>Cross-Pollination</td>
<td>32 km</td>
<td>42 m/s</td>
</tr>
</tbody>
</table>

3.5.4 Multiple Sensor Sequential Bearings

We have applied cross-pollination to two situations where the core requirement was easily satisfied—likelihood functions could be immediately applied to particles. This was due to fusing three measurements at a single epoch in Section 3.5.1 and utilizing a deterministic motion model in Section 3.5.3. Not all situations will follow these formats. In this application we demonstrate an adaptation of the cross-pollination ideas for a sequential smoothing problem with a stochastic motion model.

Say we desire a smoothing distribution $P(x_{1:6} \mid y_{1:6})$ for a tracking problem where an object follows stochastic dynamics. Furthermore, suppose that the estimates obtained are from processes ran on two sensors with differing cadences so that we desire to fuse the trajectories (particles) of $\pi_1^N$ corresponding to $P(x_1, x_3, x_5 \mid D_1 = (y_1, y_3, y_5))$ and $\pi_2^N$ corresponding to $P(x_2, x_4, x_6 \mid D_2 = (y_2, y_4, y_6))$. If the $y_i$ took place at distinct times $t_i$ and the times are increasing $t_1 < t_2 < t_3 < \ldots$, it is clear that a likelihood that corresponds to $t_2$ can not be applied to a trajectory from $\pi_1^N$ as the state at time $t_2$ has not been recorded. We note that this corresponds to the problem of having to incorporate out-of-sequence measurements (OOSM) which has been discussed in tracking and data fusion literature [3]. An attractive solution to implement cross-pollination would be to consider the Bayesian updates used in the OOSM particle filter [49]. In the following example we mitigate this problem by saving the state of the particles where any observation occurred or will occur. Depending on sensor architecture this may not be possible and other mentioned methods could be employed so that likelihood functions could be applied at all necessary times.

We apply cross-pollination to a bearings-only tracking problem inspired by Gordon et al. [27]. Several
other methodologies have been developed to perform well on the bearing-only tracking problem, for example
the resample-move filter [26]. We choose this problem not to compete with the other methods but to
demonstrate the versatility of cross-pollination to a multi-sensor sequential problem with a stochastic motion
model. We will be estimating the the state vectors of positions and velocities, up to time \(t\), which will be
driven through a linear motion model with additive Gaussian noise. That is,

\[
\theta_t = (x_1, \ldots x_t), \quad x_j = \Phi x_{j-1} + \Gamma w_j,
\]

(3.27)

\[
x_j = (x, \dot{x}, y, \dot{y})_j^T, \quad w_j = (w_x, w_y)_j^T, \quad w_j \sim N(0, \sigma_q^2 \mathbf{I}_2),
\]

(3.28)

\[
\Phi = \begin{bmatrix}
    1 & 1 & 0 & 0 \\
    0 & 1 & 0 & 0 \\
    0 & 0 & 1 & 1 \\
    0 & 0 & 0 & 1
\end{bmatrix}, \quad 
\Gamma = \begin{bmatrix}
    0.5 & 0 \\
    1 & 0 \\
    0 & 0.5 \\
    0 & 1
\end{bmatrix}.
\]

(3.29)

Bearing observations of the process will be obtained through two different sensors with additive Gaussian
noise. The \(j\)th sensor is located at \((x_{S_j}, y_{S_j})\) (actual locations \((-1, 1)\) and \((1, -1)\)) so that likelihood functions
of the measurements applied to particles are

\[
g_j(\theta^{(i)}) = \mathcal{N} \left( H(\theta^{(i)}); z_j, \sigma_r^2 \right), \quad H(\theta^{(i)}) = \arctan \left( \frac{y_i - y_{S_j}}{x_i - x_{S_j}} \right).
\]

(3.30)

The standard deviations of the sensor and model noise are \(\sigma_r = 0.005\) and \(\sigma_q = 0.001\) respectively.
The prior for initial state is a multivariate Gaussian such that

\[
x_0 \sim \mathcal{MVN}(x_p, \Sigma_p), \quad x_p = [0, 0, 0.4, -0.05]^T, \quad \Sigma_p = \begin{bmatrix}
    0.5^2 & 0 & 0 & 0 \\
    0 & 0.005^2 & 0 & 0 \\
    0 & 0 & 0.3^2 & 0 \\
    0 & 0 & 0 & 0.01^2
\end{bmatrix}.
\]

(3.31)

The initial state is \(x_0 = [-0.5, 0.001, 0.7, -0.55]^T\). This starting state is propagated through the motion
model for 20 time steps to form the true trajectory.
To demonstrate further flexibility of the ideas of cross-pollination we partition the data completely so that $D_j = y_j$ with $D = (y_1, y_2, \ldots, y_{20})$. We obtain the sample trajectories $\theta_j^{(i)}$ that comprise the empirical measure $\pi_{j,0}^N$ corresponding to $P(\theta_j \mid D_j = y_j)$ by propagating prior samples forward until time $j$, saving the state at each time $(1, 2, \ldots, j)$, and then performing importance sampling and resampling of the whole trajectory based on the $j$th likelihood function.

To start the process of sequential cross-pollination we begin with $\pi_{1,0}^N$ and motion the trajectories to $t_2$. The forward motioned trajectories of $\pi_{1,0}^N$ are then fused with the trajectories of $\pi_{2,0}^N$ by weighting the motioned trajectories with the likelihood $g_2$ and weighting the trajectories of $\pi_{2,0}^N$ with the “backward in time” likelihood $g_1$. After the cross-epoch weighting the trajectories can be pooled and resampled using norming-apart or norming-together processes as they have seen all the relevant statistical information up to the second epoch. To continue the process the cross-pollinated trajectories denoted $\pi_{2,*}^N$ are motioned forward to $t_3$ and then cross-pollinated with $\pi_{3,0}^N$. Notice that the trajectories of $\pi_{3,0}^N$ would now be weighted with two likelihoods $g_1$ and $g_2$. Extending this notion, the samples of $\pi_{j,0}^N$ would be weighted with a product of $j-1$ likelihoods. This process continues until the desired time, here $t = 20$, so that $\pi_{20,*}^N$ would estimate the measure that corresponds to $P(\theta_{20} \mid D = y_{1:20})$. Pseudo-code for this process is provided in Section 3.7. Due to particle sparsity issues raised by the application of several backward in time likelihoods we implemented this process with log likelihoods and resampled the particles during the process of applying the cross-epoch weight whenever the effective sample size was below a threshold and perturbed the particles that survived the resampling. Notice that if the initial particles are obtained via importance sampling resampling and the resample step accepts only particles originating from the measure $\pi_{j,*}^N$ at each step then this algorithm is essentially the sequential importance resampling particle filter [27]. The additional steps provide new possible trajectories that could help reduce particle sparsity.

In Figure 3.6 we see wide swaths of initial particles that comprise empirical measures at particular epochs $\pi_{j,0}^N (j = 1, 4, 7, 10, 13, 16, 19)$. The sequential cross-pollination process ends with particles that are in line with the bearing measurements for every epoch with significantly reduced uncertainty.
Figure 3.6: Sequential cross-pollination applied to a two sensor bearing-only tracking problem. Depicted are the sensor locations (purple and blue diamonds), initial particles (purple and blue circles, color matched to the sensor from which the particles have seen data from), cross-pollinated particles (black circles), and true locations (red diamond) for epochs 1, 4, 7, 10, 13, 16, 19, and 20. Time progresses from top to bottom so that epoch 1 is at the top of the figure and epoch 20 is at the bottom.

3.6 Discussion

We have presented two methods capable of fusing estimates with guarantees in the particle limit. Applying the methods to different examples have led us to some empirical insights about the methods. In our experience, norming-together is more robust to particle degeneracy issues. For example, consider the
scenario of fusing two batches of particles from two different sensors where one sensor has far more uncertainty than the other. Applying norming-apart with too few particles can be detrimental in this scenario if the uncertainty from the “bad” sensor was so wide that a single particle receives high weight. When pooled together with reasonably weighted particles from the other set this “best of the bad” particle would be replicated many times in the resampling framework and could lead to biased estimates. Computations should be performed with sufficient numbers of particles so that this problem is less likely to occur. In this same scenario it may be the case that norming-together only selects particles from the sensor with the reduced uncertainty in the resampling step, while still refining the estimates. This demonstrates that information from more uncertain sensors is still valuable and can be used in improving estimates through cross-pollination. The sequential cross-pollination example demonstrated applicability of cross-pollination ideas to stochastic motion models, however we believe that cross-pollination is most readily applicable in data-starved situations which would correspond to fewer applications of likelihood functions.

Many ideas must be considered in order to improve the method. Different weighting and resampling schemes for multiple importance sampling lead to improved variances of the estimators [18,19]. For large sets of data the weights obtained from cross-epoch weighting may render many particles degenerate. Sequential resampling may be used to combat this degeneracy so that fewer likelihoods are considered at a time [27]. In the batch case, this leads to questions about the order in which the likelihoods should be applied to the weights. For instance, if the order of application of likelihoods is based from the most to least uncertain data improvements may be possible [50].

We are interested in applications and extension of cross-pollination to the areas of multiple target tracking, distributed sensing, and optimal control. In multiple target tracking, track estimates may be cross-pollinated if they are associated to the same object. As both methods pool the sets of particles together, they could be considered centralized fusion methods. Norming-apart is decentralized before the particles are pooled together—only the likelihoods from the other are required to update the individual sets of particles for an accurate posterior estimates. The ideas in the norming-apart approach could have interesting applications in distributed sensing. Rather than passing state vectors or model parameters from sensor to sensor, likelihood functions could be passed between sensors as a way to refine estimates. This
is especially interesting in the case of limited communication bandwidths between sensors. For optimal control, it is possible to identify optimal paths to reach a specified target and associated optimal actions from Monte Carlo sampling of paths under an uncontrolled (i.e., open loop) stochastic motion model [32].

In some application instances, information about candidate paths can be available from different sensors or agents, which might then be combined with cross-pollination resampling to enable optimal control solutions with greater computational efficiency.

3.7 Algorithms

Algorithm 3 Norming-Apart Cross-Pollination

Input: Acquire the initial samples that comprise $\pi_{j,0}^N$.

for $j = 1$ to $M$ do

Obtain $\theta_j^{(i)} (i = 1, \ldots, N)$.

end

Cross-Epoch Weighting: Form $\pi_j^N$.

for $j = 1$ to $M$ do

Calculate weights $w_j^{(i)} (i = 1, \ldots, N)$ according to equation (3.3).

end

Pool Particles and Weights: Form $\pi_{MN}^N$.

Form $T = \bigcup_{j=1}^M \bigcup_{i=1}^N \theta_j^{(i)}$ and $W = \bigcup_{j=1}^M \bigcup_{i=1}^N \left( \frac{1}{N} w_j^{(i)} \right)$.

Resample: Form $\pi_N^N$.

Perform multinomial sampling on the set of weights, $W$, to obtain $N$ (from $M \times N$ total) particles $\theta^{(i)} (i = 1, \ldots, N)$ from $T$. 

Algorithm 4 Norming-Together Cross-Pollination

Input: Acquire the initial samples that comprise $\pi_{j,0}^N$.

for $j = 1$ to $M$ do

| Obtain $\theta_j^{(i)}$ ($i = 1, \ldots, N$) |
| end |

Cross-Epoch Weighting: Form $\pi_j^N$.

for $j = 1$ to $M$ do

| Calculate weights $w_j^{(i)}$ ($i = 1, \ldots, N$) according to equation (3.7). |
| end |

Pool Particles and Weights: Form $\pi_{1}^{MN}$.

Form $T = \bigcup_{j=1}^{M} \bigcup_{i=1}^{N} \theta_j^{(i)}$ and $W = \bigcup_{j=1}^{M} \bigcup_{i=1}^{N} w_j^{(i)}$.

Resample: Form $\pi^N$.

Perform multinomial sampling on the set of weights, $W$, to obtain $N$ (from $M \times N$ total) particles $\theta^{(i)}$ ($i = 1, \ldots, N$) from $T$.

Algorithm 5 Sequential Cross-Pollination

Initialization:

Obtain $\theta_1^{(i)}$ ($i = 1, \ldots, N$) that comprise $\pi_{1,0}^N$ corresponding to $P(\theta_1 | z_1)$

Set $\theta_{1,*}^{(i)} = \theta_1^{(i)}$

Iteration Cross-Pollination:

for $j=2$ to $M$ do

| Obtain $\theta_j^{(i)}$ ($i = 1, \ldots, N$) that comprise $\pi_{j,0}^N$ corresponding to $P(\theta_j | z_j)$ |
| Compute unnormalized weights $\omega_j^{(i)} = \prod_{k<j} g_k \left( \theta_j^{(i)} \right)$ |
| Motion $\theta_{j-1,*}$ forward to $t_j$ and set $w_j^{(i)} = g_j \left( \theta_j^{(i)} \right)$ |
| Normalize weights, pool particles and weights, and resample particles as in 3 or 4 |
| Set result equal to $\theta_{j,*}^{(i)}$ |
| end |
Chapter 4

Cross Entropy Population Monte Carlo

This chapter presents a paper I wrote with Dr. Jem Corcoran and Dr. Michael Schneider which can be found at https://arxiv.org/abs/2110.05684. This work has been presented at seminars at Lawrence Livermore National Lab.

4.1 Contribution

We present a novel Cross-Entropy based population Monte Carlo algorithm. Using the cross-entropy method as a means of updating the population of proposal distributions make this method directly applicable to rare event problems which is not the case for the majority of population Monte Carlo algorithms. For instance, as explained in Section 4.6, to make the the local and global resampling methods suited to rare event problems we choose to include the performance function in the importance weights. The closest relatives of this work involve using the cross-entropy method and similar entropy methods to update the parameters of mixture distributions. When updating mixture distributions the update equations become entangled between the mixture components. Our reinterpretation of the cross-entropy method in Section 4.5 frees the update equations from this entanglement resulting in individual updates for each proposal distribution. Furthermore, we discuss that scheduling covariance updates after mean updates was important to success. We show efficacy on various numerical examples and demonstrate the algorithm on a conjunction analysis problem.
4.2 Introduction

Rare events are events that happen with very low frequency. While the definition of low frequency is domain specific, the terminology is typically reserved for events deemed disruptive and even catastrophic, in areas such as such as in structural reliability [35], conjunction assessment [39], climate modeling [62], and epidemiology [40]. Estimating rare event probabilities using Monte Carlo techniques is computationally expensive, often to the point of intractability, and special techniques are required [53]. Such techniques include subset splitting, line sampling, and importance sampling. In the case of importance sampling, a proposal distribution must be chosen by the user, and a poor choice will have the undesirable effect of producing high variance estimates. Adaptive importance sampling (AIS) algorithms allows one to update the proposal distribution based on intermediate results. A well known AIS algorithm is the cross-entropy algorithm of Rubenstein and Melamed [54, 55] which aims to minimize the Kullback–Leibler divergence between the proposal distribution and the optimal sampling density from a given parametric family using incremental parameter changes.

The population Monte Carlo (PMC) algorithm [10] is an AIS algorithm that can be used for estimating rare event probabilities or, more generally, expectations with respect to a given target distribution. At each iteration, a Markov transition kernel is used to propagate a set of particles (samples) forward in time. Importance sampling weights are attached to each propagated particle and a new set of particles is sampled according to those weights. At any stage, the weighted particles can be used to give an unbiased estimator of the probability of interest. In fact, all particles and weights from all iterations can be used. In this chapter we give brief background on rare events, importance sampling, and the cross-entropy method. We then show how a reinterpretation of the derivation of the method leads naturally to a population Monte Carlo scheme and demonstrate its efficacy on several examples.
4.3 Rare Events and Importance Sampling

4.3.1 Rare Event Problem

The rare event problem involves estimating probabilities of a random variable $x$ exceeding a level $\gamma$ of a performance function $S(x)$. Assuming $x$ is distributed according to a probability distribution function $\pi(x)$ we are seeking to estimate

$$\ell = P(S(x) \geq \gamma) = E_{\pi} \left[ I \{ S(x) \geq \gamma \} \right],$$

where $I$ is an indicator function. Due to the rareness of the event standard Monte Carlo techniques require exorbitant amounts of samples to provide accurate estimates. To improve on this importance sampling techniques are often applied to find important regions, that is, regions where samples are more likely to exceed the required performance level.

4.3.2 Importance Sampling

Importance sampling (IS) is a ubiquitous Monte Carlo technique that draws samples from a proposal distribution and weights them accordingly to match expectations from a target distribution [52]. In terms of the rare event problem and given a proposal distribution with pdf $q(x)$ the IS estimator is

$$E_{\pi} \left[ I \{ S(x) \geq \gamma \} \right] \approx \hat{\ell} = \frac{1}{K} \sum_{k=1}^{K} I \{ S(x_k) \geq \gamma \} w(x_k),$$

where $x_k$ are drawn from the importance distribution $q$ and $w(x_k) = \pi(x_k)/q(x_k)$ are the importance weights. The importance weights can be interpreted as the Radon-Nikodym derivative between the measures that correspond to the pdfs $\pi$ and $q$. Central to the effectiveness of IS is the choice of the proposal distribution. Naturally, many methods have been employed to iteratively adapt the proposal distribution. Furthermore, rather than adapting a single distribution, adapting a population of distributions, $q_1, \ldots, q_N$, has spurred many AIS methods which are well-reviewed by Bugallo et al. [8]. An important advance in the weighting scheme of multiple importance sampling (MIS) is the concept of the deterministic mixture weight (DM-weight) [19]. In this weighting scheme the samples are weighted as if drawn from the mixture of distributions,
that is the importance weight for a sample is given by

\[ w(x_k) = w_k = \frac{\pi(x_k)}{\frac{1}{N} \sum_{n=1}^{N} q_n(x_k)} , \tag{4.1} \]

where we have introduced the notation \( w_k \) to mean the weight of the \( k \)th sample. The advantage of this weighting scheme is it reduces the variance of the IS estimators.

Several variations of the adapting the family of proposals have been proposed. Adaptation through resampling schemes: local resample PMC (LR-PMC) adapts a proposal in the family through multinomial resampling based on samples produced from that proposal for all proposals independently, in contrast global resample PMC (GR-PMC) adapts all proposals at once by performing multinomial sampling on all samples produced by all the proposals [18]. Gradient adaptive population importance sampler (GAPIS) utilizes the target distributions gradient and Hessian matrix to adapt proposals [20]. Utilizing popular Markov chain Monte Carlo (MCMC) techniques has also been studied recently to adapt proposals e.g. Langevin based PMC (SL-PMC) [16] and Hamiltonian AIS (HAIS) [47]. Further MCMC-driven IS techniques are discussed by Llorent et al. [38]. An important realization discussed by Llorent et al. is that it is unknown what distribution the MCMC methods should target. One way of producing the best possible distribution to perform importance sampling from is the cross-entropy method [54]. Here, the measure of best is the Kullback-Leibler divergence between the optimal importance sampling distribution and the proposal distributions. Mixture distributions have previously been adapted via the cross-entropy process [35, 61], and through an expectation-maximization like procedure in mixture PMC [9] and D-kernel PMC [15]. While the method in this chapter utilizes the mixture weights we focus on adapting individual proposals as opposed to the mixture of distributions, this eliminates update equations for the weights of the mixture. As discussed below, the method in this chapter involves gradients with respect to the parameters of the proposal distributions, which stands apart from GAPIS, SL-PMC, and HAIS which require gradients of the logarithm of the target distribution. This is especially helpful in the case of rare events which are often posed in the form of satisfying a performance or indicator function, which would necessarily create discontinuities in the gradient of the target distribution.
4.4 Cross-Entropy Method

The cross-entropy method is often utilized for rare-events and combinatorial optimization tasks. In particular we utilize the multi-level algorithm as described by DeBoer et al. [13]. An importance sampling proposal distribution, indexed by parameters $\mu$, is selected, and the optimal parameter is sought through incremental changes. The algorithm proceeds in two stages, involving updating temporary performance levels $\gamma_t$ to build up to the desired performance $\gamma$ and then updating the parameters of the proposal from $\mu^{(t)}$ to $\mu^{(t+1)}$. From step $t$ to step $t+1$ of the algorithm, samples $\{x_k\}_{k=1}^K$ are obtained from the proposal distribution $q(\cdot; \mu^{(t)})$, the performance function is evaluated on the samples, these performances are then sorted and the $(1 - \rho)$ sample quantile of the performances is used to determine the temporary performance level. Samples that provide performance beyond the temporary performance level are then used to update the parameters to minimize the Kullback Leibler divergence between the optimal sampling density and proposal distribution, this results in optimization problem

$$\max_{\mu} \frac{1}{K} \sum_{k=1}^K \mathbb{I}(S(x_k) \geq \gamma_t) \frac{\pi(x_k)}{q(x_k; \mu^{(t)})} \ln q(x_k; \mu^{(t)}).$$

(4.2)
Algorithm 6 Cross-Entropy Algorithm

Input: Quantile parameter $\rho$, initial proposal parameter $\mu^{(0)}$, number of values to be sampled $K$.

Initialization Set $t = 0$ and set $\hat{\gamma}_0$ to be any value below $\gamma$.

while $\hat{\gamma}_t < \gamma$ do
  Set $t = t + 1$.

  sampling: Draw $K$ samples from the importance sampling density
  
  \[
  \{x_k^{(t)}\}_{k=1}^K \sim q(\cdot; \mu^{(t-1)}).
  \]

  performances: Evaluate the performance function $S(x)$ at each sampled value and order the results to produce $S(1) \leq S(2) \leq \cdots \leq S(K)$.

  sample quantile: Let $\hat{\gamma}_t = S(\lceil (1-\rho)K \rceil)$ be the $(1-\rho)$ sample quantile of the performances.

  if $\hat{\gamma}_t < \gamma$ then
    Solve the CE Update Equation (4.2) to obtain $\mu^{(t)}$.
  else
    Set $\hat{\gamma}_t = \gamma$.
  end

end

Output: $\hat{\ell} = \frac{1}{K} \sum_{k=1}^K I_{\{S(x_k^{(t)}) \geq \gamma\}} \frac{\pi(x_k^{(t)})}{q(x_k^{(t)}; \mu^{(t)})}$.

4.5 Cross-Entropy Population Monte Carlo

To incorporate the cross-entropy method as a way to update the parameters for the population of proposal distributions, we modify the typical derivation of the cross-entropy method, starting with the minimization of Kullback-Leibler divergence between the optimal distribution $p(x) = \frac{1}{\pi} I_{\{S(x) \geq \gamma\}} \pi(x)$ to create a new stochastic program to optimize the family of proposals, briefly let $Q_\mu$ denote the family of proposals as a mixture distribution,
\[
\min_{\mu} KL(p \| Q_\mu) = \max_{\mu} \int p(x) \ln (Q(x; \mu)) \, dx
\]

\[
\approx \max_{\mu} \frac{1}{NK} \sum_{n=1}^{N} \sum_{k=1}^{K} I(S(x_{n,k}) \geq \gamma) \frac{\pi(x_{n,k})}{N} \ln (Q(x_{n,k}; \mu_n))
\]

\[
= \max_{\mu_n} \frac{1}{N} \sum_{n=1}^{N} \sum_{k=1}^{K} I(S(x_{n,k}) \geq \gamma) \frac{\pi(x_{n,k})}{N} \ln (q_n(x_{n,k}; \mu_n))
\]

The first approximation is the unbiased multiple importance sampling approximation with the DM-weights taken from the previous trial. The first equality comes from knowing that the \(n\)th sample is drawn from the \(n\)th proposal distribution, so the pdf applied to that sample is \(q_n\). Thus the stochastic program splits into \(N\) individual optimization problems – one optimization problem for each proposal distribution. That is for \(n = 1, \ldots, N\) do the following optimization

\[
\max_{\mu_n} \frac{1}{K} \sum_{k=1}^{K} I(S(x_{n,k}) \geq \gamma_n) \frac{\pi(x_{n,k})}{N} \ln (q_n(x_{n,k}; \mu_n))
\] (4.3)

Intuitively, the parameters of every proposal distribution are updated in a cross-entropy multilevel fashion with the DM-weights in place of the typical IS weights. Heuristically, the denominator in the DM-weights promotes distance between proposals distributions as samples from a particular proposal distribution will be weighted more heavily when further away (less probable) from the other proposals. In the algorithm below, we adopt the typical set-up for most PMC methods– that is we set a number of proposals \(N\), number of samples \(K\), and number of trials \(T\). Fixing the number of samples and trials is not required of the CE-PMC method. In some applications of the cross-entropy method rounds of presampling with a smaller number of samples are used to find the importance region followed by a final trial with a larger number of samples. Instead of a set number of trials, many stopping criteria could be employed e.g the one used in Algorithm 6–stopping when the sample quantile exceeds the desired performance.
Algorithm 7 Cross-Entropy Population Monte Carlo

**Input:** Quantile parameter $\rho$, number of proposals $N$, number of samples per proposal $K$, number of trials $T$, and parameters of initial distributions $\{\mu_n^{(1)}\}_{n=1}^N$.

for $t = 1, \cdots, T$ do

    for $n = 1, \cdots, N$ do

        **sampling:** Draw $K$ samples from each proposal

        $$\{x_{n,k}^{(t)}\}_{k=1}^K \sim q_n(\cdot; \mu_n^{(t)}).$$

        **weighting:** Weight every sample with the DM-weight

        $$w_{n,k}^{(t)} = \frac{\pi(x_{n,k}^{(t)})}{\frac{1}{N} \sum_{n=1}^N q_n(x_{n,k}^{(t)}; \mu_n^{(t)}).}$$

    end

for $n = 1, \cdots, N$ using $\{x_{n,k}^{(t)}, w_{n,k}^{(t)}\}_{k=1}^K$ do

    **performances:** Obtain $S_{(i)}$ see Algorithm 6.

    **sample quantile:** Obtain $\hat{\gamma}_t$, see Algorithm 6.

    **adapting:** Solve CE Update (4.3) to obtain $\mu_n^{(t+1)}$

end

**Output:** All samples and their respective weights

$$\{x_{n,k}^{(t)}, w_{n,k}^{(t)}\}_{n=1,k=1,t=1}^{N,K,T}$$

end

In AIS methods typically the final estimator is given by

$$\hat{\ell} = \frac{1}{T NK} \sum_{t=1}^T \sum_{n=1}^N \sum_{k=1}^K I_{S(x_{n,k}^{(t)} \geq \gamma)} w_{n,k}^{(t)}.$$  

As the cross-entropy method should converge to the optimal parameters, we will only consider the estimate given by the samples produced in the final trial as in the output of Algorithm 6.
4.6 Examples

For each experiment we will adapt a family of Gaussian distributions and compare the LR-PMC, GR-PMC, and CE-PMC methods. When updating a single Gaussian distribution mean $\mu$ and covariance $\Sigma$ the update equations are given by

$$
\mu = \frac{\sum_{k=1}^{K} I\{S(x_k) \geq \hat{\gamma}_t\} w_k x_k}{\sum_{k=1}^{K} I\{S(x_k) \geq \hat{\gamma}_t\} w_k}
$$

$$
\Sigma = \frac{\sum_{k=1}^{K} I\{S(x_k) \geq \hat{\gamma}_t\} w_k (x_k - \mu)(x_k - \mu)^T}{\sum_{k=1}^{K} I\{S(x_k) \geq \hat{\gamma}_t\} w_k}
$$

where we take $w_k$ to be the deterministic mixture weights, produced by the previous trial.

Although typically, cross-entropy updates are run until a threshold is met, for a fair and consistent evaluation between the three methods, we fix the number of trials $T$, the number of proposals $N$, and the number of samples per proposal $K$ for each example. Furthermore, we impose a few ad hoc conventions. To make LR-PMC and GR-PMC methods able to adapt to the rare event set-up we include the indicator function in the importance weight, as suggested by Elvira et al [21, Eq 5.1]. Proposals can often produce sets of samples all with zero weight, thus halting any effort to perform a multinomial resample. For LR-PMC the convention when a proposal produced all $K$ samples with zero weight, was to reweight the samples evenly with weight $1/K$, and proceed with the algorithm. For GR-PMC, the multinomial resampling breaks down if all $NK$ samples have zero weight, in which case we reweight all the samples evenly with weight $1/NK$ and proceed as the algorithm intended. In CE-PMC when updating the covariances of a Gaussian distribution one or several dimensions may flatten resulting in a singular matrix, especially when approaching a linear function [25, Sec. 6.3]. We implement two operational procedures to mitigate this problem, the first is to check if the updated covariance matrix is singular, if it is we adopt the covariance from the previous trial, the second procedure is to only utilize the mean updating formula in the first half of the trials, and update both the mean and covariance in the second half of the trials, this is a version of scheduling covariance updates [10, Sec. 3]. More advanced techniques may be implemented to fix this problem, for instance, the modified metropolis algorithm of subset simulation [1] could be used to expand the covariance in the
directions of decay.

### 4.6.1 Structural Reliability Examples

First we examine three examples taken from structural reliability literature [25, 35]. The target distributions are all proportional to $I_{\{S_i(x) \leq \gamma\}} \pi(x)$, where $\pi(x) = \pi(x_1, x_2)$ is given by a standard multivariate Gaussian distribution, and

$$S_1(x) = 5 - x_2 - 0.5(x_1 - 0.1),$$

$$S_2(x) = 5 - x_2 - 0.1(x_1),$$

$$S_3(x) = \min \begin{cases} 
3 + (x_1 - x_2)^2/10 - (x_1 + x_2)/\sqrt{2} \\
3 + (x_1 - x_2)^2/10 + (x_1 + x_2)/\sqrt{2} \\
x_1 - x_2 + 7/\sqrt{2} \\
x_2 - x_1 + 7/\sqrt{2}
\end{cases}.$$  

We refer to the problems respectively as S1, S2, and S3 the problems have the respective reference values $3.01e-3$, $8.67e-7$, and $2.22e-3$. We compare the performance of LR-PMC, GR-PMC, and CE-PMC on these three examples, with $N = 25$, $K = 100$, and $T = 20$ and average the results over 1000 runs. As shown in Table 4.1, the CE-PMC outperformed the other methods with respect to relative root mean squared error (RRMSE). In Figure 4.1 we see that the CE-PMC algorithm is able to match complex regions of importance to produce reliable estimates.
Table 4.1: RRMSE on Structural Reliability Examples

<table>
<thead>
<tr>
<th>Method/Problem</th>
<th>$S_1$</th>
<th>$S_2$</th>
<th>$S_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR-PMC</td>
<td>0.0424</td>
<td>0.0602</td>
<td>0.0542</td>
</tr>
<tr>
<td>GR-PMC</td>
<td>0.0602</td>
<td>0.0494</td>
<td>0.6603</td>
</tr>
<tr>
<td>CE-PMC</td>
<td>0.0163</td>
<td>0.0141</td>
<td>0.0233</td>
</tr>
</tbody>
</table>

Figure 4.1: Top: Contour plots of the desired distribution for $S_1$ (left), $S_2$ (middle), and $S_3$ (right). Bottom: Example of contour plots of the estimated distribution produced by CE-PMC.
4.6.2 Variable Dimension Problem

We consider the problem $S_4(x) = \beta - \frac{1}{\sqrt{D}} \sum_{i=1}^{D} x_i$ where $x_i$ are drawn from standard normal distributions [35]. This problem is particularly interesting as the probability is $\Phi(-\beta)$ regardless of dimension $D$, where $\Phi$ is the cumulative distribution function of the standard normal. This allows for testing the methods response to changes in dimension. We set $\beta = 5$ so that the true rare event probability is $\Phi(-5) \approx 2.86e-7$ and run 100 independent tests with $N = 4, K = 5000, T = 32$ for each dimensions 2, 5, 10, 20, 30, 40, and 50. The initial means of the distributions were chosen by scaling, between $-1$ and $1$, centered Latin hypercube samples [43] i.e. every entry of $\mu^{(0)}$ is either $\pm 0.25$ or $\pm 0.75$. The initial covariances were isotropic, $\Sigma^{(0)} = \sigma^2 I_D$ with $\sigma = 1$. The results of this experiment are displayed in the Figure 4.2 It is clear that CE-PMC performs better than LR-PMC and GR-PMC as the dimension increases.

![Figure 4.2: Mean Probability over 100 Monte Carlo runs against increasing dimension $D$ for the variable dimension problem with $\beta = 5$.](image-url)
4.6.3 Conjunction Analysis

A problem of much importance in the space domain community is that of conjunction analysis, which involves finding the probability of an object in space passing nearby another object e.g. satellites and space debris. Consider a rogue object with Gaussian uncertainty with covariance (5 meters in position and 0.1m/s in velocity) at time $t_0$ (denoted the distribution by $\pi$) and Cartesian position denoted by $r$, we are concerned with the probability that this rogue object comes within 50 meters at time $t_1 \approx t_0 + 9893.34$ s of two assets with position denoted by $a_1$ and $a_2$ which are on the same orbital plane 10,000 meters apart at $t_0$. We simulate the positions and velocities of all objects with Keplerian propagators. Our performance function is $S(x) = \min_i ||r-a_i||^2$ and our goal is to estimate the probability that the rogue object comes within 50 meters of the assets, $\ell = \mathbb{E}_\pi [I_{\{S(x)\geq 50\,\text{m}\}}]$. Performing the CE-PMC algorithm with $N = 16$ Gaussian distributions, $K = 500$ samples per distribution, and $T = 20$ trials we estimate $\hat{\ell}_{CE} \approx 0.000113$. To confirm the validity of this result we simulate with 1 million Monte Carlo samples and get the estimate $\hat{\ell}_{MC} = 0.000111$. In Figure 4.3 we again see the ability of CE-PMC to find the regions of importance, the curtain of naive Monte Carlo samples are refined to threads of importance by application of the CE-PMC algorithm.
Figure 4.3: Traces of orbital position of samples (red) and assets (blue). The top portion has samples derived from naive Monte Carlo samples. The bottom portion has samples derived from CE-PMC. The plots on the left depict the traces from $t_1 - 500$ to $t_1 + 50$. The plots on the right depict the traces from $t_1 - 5$ to $t_1 + 5$. 
4.6.4 Revisiting Data Fusion

In this example we revisit the problem of orbital data fusion problem of Section 3.5.3 and examine the applicability of CE-PMC. As the CE-PMC algorithm is suited to the rare events we consider the following idea: satisfying a product of likelihoods can be seen as a rare event when samples are sourced from a prior distribution. Given likelihood functions $g_j(\theta)$ for $j = 1, \ldots, M$ we can run the CE-PMC algorithm to target samples that perform well on a function proportional to sum of the log likelihood functions, $S(\theta) = \frac{1}{C} \sum_{j=1}^{M} \ln(g_j(\theta))$.

Looking at the typical derivation of the self-normalized importance sampler we have,

$$E_{p(\theta|y)} [h(\theta)] = \int h(\theta)p(\theta|y) \, d\theta$$
$$= \int \frac{h(\theta)p(y|\theta)p(\theta)}{p(y)} \, d\theta$$
$$= \frac{\int h(\theta)p(y|\theta)p(\theta) \, d\theta}{\int p(y|\theta)p(\theta) \, d\theta}$$
$$= \frac{E_{p(\theta)} [h(\theta)p(y|\theta)]}{E_{p(\theta)} [p(y|\theta)]}$$
$$\approx \sum_{i=1}^{K} \bar{w}_k h(\theta_k), \quad \theta_k \sim p(\theta)$$

where $\bar{w}_k$ are the normalized importance weights. Incorporating the proposal distributions from the final round of CE-PMC would give the estimate

$$E_{p(\theta|y)} [h(\theta)] \approx \frac{\sum_{n=1}^{N} \sum_{k=1}^{K} h(\theta_{T_n,k}) p(y|\theta_{T_n,k})}{\sum_{n=1}^{N} \sum_{k=1}^{K} p(\theta_{T_n,k})} \frac{p(\theta_{T_n,k})}{\sum_{n=1}^{N} q_\alpha(\theta_{T_n,k})}.$$ 

As the CE-PMC method is seeking better performances of $S(x)$, care must be put into the choice of $\gamma$ and stopping conditions. For instance, looking at an orbital determination problem similar to Section 3.5.3, where we consider 6 right ascension and declination measurements taken by two different observers in LEO with Gaussian likelihoods both with isotropic covariance $\sigma^2 I_2$ with $\sigma = 20$ arcseconds, which we denote as function of $\theta$ by $g_j(\theta)$. Given that the true orbit has position $r_t$ and velocity $v_t$ at time $t_0$, we set the prior distribution to be Gaussian with positional mean $1.1 r_t$ and velocity mean $0.9 v_t$ and covariance with entries...
along the diagonal of \((.1 \, R_{GEO})^2\) and \((.1V_{GEO})^2\) for the position and velocity dimensions respectively, 
where \(R_{GEO} \approx 42164172 \, m\) and \(V_{GEO} \approx 3074 \, m/s\). We run the CE-PMC algorithm with \(N = 2\), \(K = 1000\), 
and \(T = 50\) with Gaussian distributions of with means sampled from the prior and diagonal covariance with 
entries along the diagonal of \((.05 \, R_{GEO})^2\) and \((.05 \, V_{GEO})^2\). We set \(S(x) = \frac{1}{10000} \sum_{j=1}^{6} \ln(g_j(\theta))\) We see the 
need for careful set-up of the method displayed in Figure 4.4, with no stopping criteria other than reaching 
the end of trials we have essentially produced a maximum likelihood estimate, which is to say we have 
seemingly over fit the data. To advance this method to be more applicable, stopping criteria and desired 
performance level should be investigated.

![Figure 4.4: Trace of the true orbit (blue) and 100 samples from the final trial of the CE-PMC algorithm (red).](image)

**4.7 Conclusions and Future Directions**

When running experiments the schedule of updating the means in the first half of trials, followed by 
means and covariances in the following trials, was important to the performance of the algorithm, without 
this process the high-dimensional examples proved to be very challenging.

There are many interesting avenues that can be pursued in the future. The cross entropy population 
Monte Carlo algorithm is uniquely suited for success on rare event problems, adapting other population
Monte Carlo methods for performance could be an interesting pursuit. As performed in our numerical examples and suggested by Elvira et al. [21] to make the resampling strategy PMC algorithms viable we had to incorporate the use of the indicator function into the importance weights, to make use of PMC methods that use gradients of the target distribution it may be necessary to consider how smoothing the performance functions as well as including the performance function in importance weights could allow for increased performance. There are interesting paths of future work in the area of path integral optimal control. Kappen et al. [33] have devised formulas for optimal distributions of trajectories and applied an importance sampling cross-entropy method to return optimal controls. Greater exploration of the parameter space of the control could be achieved via CE-PMC.
Bibliography


