http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

RESIDUAL AND STRATIFIED BRANCHING PARTICLE FILTERS

BY MICHAEL A. KOURITZIN

University of Alberta

ABSTRACT. A class of discrete-time branching particle filters is introduced with individual resampling: If there are $\mathbb{N}_n$ particles alive at time n, $\mathbb{N}_0 = N$, $a_n \leq 1 \leq b_n$, $\widehat{\mathbb{L}}_{n+1}^i$ is the current unnormalized importance weight for particle i and $\mathbb{A}_{n+1} = \frac{1}{N}\sum_{i=1}^{\mathbb{N}_n} \widehat{\mathbb{L}}_{n+1}^i$, then weight is preserved when $\widehat{\mathbb{L}}_{n+1}^i \in (a_n\mathbb{A}_{n+1}, b_n\mathbb{A}_{n+1})$. Otherwise, $\left\lfloor \frac{\widehat{\mathbb{L}}_{n+1}^i}{\mathbb{A}_{n+1}} \right\rfloor + \rho_n^i$ offspring are produced and assigned weight $\mathbb{A}_{n+1}$, where ρ_n^i is a Bernoulli of parameter $\frac{\widehat{\mathbb{L}}_{n+1}^i}{\mathbb{A}_{n+1}} - \left\lfloor \frac{\widehat{\mathbb{L}}_{n+1}^i}{\mathbb{A}_{n+1}} \right\rfloor$. The algorithm is shown to be stable with respect to the number of particles. Its performance is significantly better than the popular bootstrap algorithm as well as the residual, stratified and systematic resampled variants. Moreover, our branching filters run much faster than these other particle filters, especially when performing Bayesian model selection, wherein one not only needs to track but also to select which model best represents the data. Still, on tracking alone, our best branching filters have approximately 200 times the power of the bootstrap algorithm on our problems.

1. INTRODUCTION

Nonlinear filtering deals with determining the distribution of the current state of a non-observable, random, dynamic signal X given the history of a distorted, corrupted partial observation process Y living on the same probability space $(\Omega, \mathcal{F}, P)$ as X. Bayesian model selection, sometimes done while filtering, deals with determining which of a class of signal models $\{X^{(i)}\}_{i\in I}$ best fits the observed values of Y by pairwise Bayes' factor comparison. For many practical problems each potential signal is a time-homogeneous discrete-time Markov process $\{X_n,\ n = 0, 1, 2, ...\}$, living on some complete, separable metric space (E, ρ), with initial distribution π_0 and transition probability kernel K. The observation process takes the form ($Y_0 = 0$ and) $Y_n = h(X_{n-1}) + V_n$ for $n \in \mathbb{N}$, where $\{V_n\}_{n=1}^{\infty}$ are independent random vectors with common *strictly positive, bounded* density g that are independent of X, and the sensor function h is a measurable mapping from E to $\mathbb{R}^d$. Then, the objective of filtering (with respect to any given signal model X) is to compute the conditional probabilities $\pi_n(A) = P\left(X_n \in A \middle| \mathcal{F}_n^Y\right)$, $n = 1, 2, ...$, for all Borel sets

2010 *Mathematics Subject Classification.* Primary 60G35; Secondary 62M20, 60G09, 60J80.

Key words and phrases. Particle Filter, Branching Process, Bayesian Model Selection.

Partial funding in support of this work was provided by an NSERC discovery grant.
The author is indebted to Xingpu Wang for his help with the simulations.

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

A or, equivalently, the conditional expectations $\pi_n(f) = E^P\left(f(X_n)\middle|\mathcal{F}_n^Y\right)$ for all bounded, measurable functions $f : E \to \mathbb{R}$, where $\mathcal{F}_n^Y \doteq \sigma\{Y_l, l = 1, ..., n\}$ is the information obtained from the back observations. On the other hand, the objective of Bayes' factor model selection is to compare the ratio $B_n^{12} = \frac{E^Q[L_n(Y|X^{(1)})|\mathcal{F}_n^Y]}{E^Q[L_n(Y|X^{(2)})|\mathcal{F}_n^Y]}$ of marginal likelihoods between potential signal models $X^{(1)}$ and $X^{(2)}$ with respect to some reference probability measure Q. (The metric space E, initial distribution π_0, transition probability K and sensor function h can all depend upon the potential signal $X^{(i)}$ as long as the observation noise $\{V_n\}$ is the same.)

Suppose without loss of generality that $\Omega = (E \times \mathbb{R}^d)^\infty$ and $\mathcal{F} = \mathcal{B}((E \times \mathbb{R}^d)^\infty)$ until later extended. Moreover, suppose hereafter $\mathcal{F}_{-1}^\xi \doteq \{\emptyset, \Omega\}$, $\mathcal{F}_n^\xi \doteq \sigma\{\xi_l^k, k \in \mathcal{K}, l \leq n\}$ when $n \in \mathbb{N}_0$ and $\mathcal{F}_\infty^\xi \doteq \sigma\{\xi_l^k, k \in \mathcal{K}, l < \infty\}$ for random variables $\{\xi_n^k, k \in \mathcal{K}, n \in \{0, 1, ...\}\}$ on $(\Omega, \mathcal{F})$. (This is consistent with $\mathcal{F}_n^Y$ defined above if $\mathcal{K}$ has one element.) For unnormalized filters, we transfer the information contained in the observations to a likelihood process by measure change. In this method, a reference probability measure Q is introduced under which the signal, observation process $\{(X_n, Y_{n+1}),\ n = 0, 1, ...\}$ has the same distribution as the signal, noise process $\{(X_n, V_{n+1}),\ n = 0, 1, ...\}$ does under P. Hence, the observations are i.i.d. random vectors with strictly positive bounded density g and are independent of X under measure Q. All the observation information is absorbed into the likelihood process $\{L_n,\ n = 1, 2, ...\}$ transforming Q back to P, which in our case has the form

$$\left.\frac{dP}{dQ}\right|_{\mathcal{F}_\infty^X \vee \mathcal{F}_n^Y} = L_n = \prod_{j=1}^n \alpha_j(X_{j-1}), \text{ with } \alpha_j(x) = \frac{g(Y_j - h(x))}{g(Y_j)}, \tag{1.1}$$

so $L_n = \alpha_n(X_{n-1})L_{n-1}$ and $L_0 = 1$. The following (well-known) discrete Girsanov's theorem constructs the real probability P from the reference Q.

Theorem 1. *Suppose that $\{X_n,\ n = 0, 1, ...\}$ and $\{Y_n,\ n = 1, 2, ...\}$ are independent processes on $(\Omega, \mathcal{F}, Q)$, the $\{Y_n\}$ are i.i.d. with strictly-positive, bounded density g on $\mathbb{R}^d$ and $V_n \doteq Y_n - h(X_{n-1})$ for all $n = 1, 2, ...$ Then, there exists a probability measure P such that (1.1) holds, $\{V_n,\ n = 1, 2, ...\}$ are i.i.d. on $(\Omega, \mathcal{F}, P)$ with density g and $\{X_n\}$ is independent of $\{V_n\}$ with the same law as on $(\Omega, \mathcal{F}, Q)$.*

Filtering and model selection can be done concurrently using *unnormalized filters*

$$\sigma_n(f) = E^Q\left(L_n f(X_n)\middle|\mathcal{F}_n^Y\right), \tag{1.2}$$

so $\sigma_0 = \pi_0$, as $L_0 = 1$ and $\mathcal{F}_0^Y = \{\emptyset, \Omega\}$. Then, the filter satisfies $\pi_n(f) = \frac{\sigma_n(f)}{\sigma_n(1)}$ by Bayes rule and the Bayes factor satisfies $B_n^{12} = \frac{\sigma_n^{(1)}(1)}{\sigma_n^{(2)}(1)}$, where $\sigma_n^{(i)}(f) = E^Q\left(L_n^{(i)} f\left(X_n^{(i)}\right)\middle|\mathcal{F}_n^Y\right)$ with $L_n^{(i)} = \prod_{j=1}^n \alpha_j(X_{j-1}^{(i)})$ is the unnormalized filter for signal model $X^{(i)}$. Therefore, we can combine Bayesian model selection *and* filtering (for each potential signal) by constructing approximations (denoted $\mathbb{S}_n^N$ below) to the unnormalized filter for each candidate signal model as done in Kouritzin and Zeng (2005a), Kouritzin and Zeng (2005b) and Kouritzin (2015).

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

Nowadays, particle filters are utilized widely in as diverse areas as econometrics, defense and clickstream analysis. The original (resampled) interacting particle filters, starting with the bootstrap filter, have been intensely studied (see e.g. Del Moral and Miclo (2000) and Cappe et. al. (2007) for an overview and historical account). However, Del Moral et. al. (2000) show that the performance of a particle filter depends heavily upon the resampling used and resampling methods are currently an active area of investigation. Furthermore, resampled particle filters approximate the actual filter π_n so prior filter estimates must be stored to perform model selection. On the other hand, the weighted particle filter (largely credited to Handschin (1970) as well as Handschin and Mayne (1969) and studied in Kurtz and Xiong (1999), Kurtz and Xiong (2000)) approximates the unnormalized particle filter σ_n, is the most basic particle filter and is embarassingly computer parallelizable. More generally, branching particle filters, like those introduced by Crisan and Lyons (1997), can have model selection capabilities, effective resampling and be highly parallelizable. However, branching particle filters suffer from dramatic particle swings and difficult analysis - or do they? Herein, we introduce and analyze branching particle filters that avoid the weighted-particle-filter particle spread problems yet still have immediate model selection capabilities. They include the weighted particle filter as the extreme zero-resampling case and a branching variation of the better algorithm in Del Moral et. al. (2000) as the extreme fully-resampled case. They are relatively stable with respect to particle number swings and can be analyzed using exchangeability (in lieu of independence).

There are many approaches to reducing resampling noise in the bootstrap filter. For example, researchers brought in importance sampling and delayed bulk resampling methods (see e.g. Del Moral et. al. (2012)). Others have introduced less noisy types of resampling, which we discuss below. However, there are few studies like Ballantyne et. al. (2000) of the practical partially-resampled algorithms where decisions are made on a particle-by-particle basis with the aim of only removing the poor particles and splitting the best particles (in an unbiased manner). Kouritzin and Sun (2005) do obtain L_2-rates of convergence for a partially-resampled algorithm in a specific setting. Our present work introduces new classes of branching particle filters, motivates their use and sets up a framework for studying them.

In the next section, we consider resampled (Bootstrap-related) particle filters on tracking and model selection problems. In particular, multinomial, residual, stratified, systematic and combined residual-stratified schemes are considered. Our class of branching particle filters is introduced and studied in Section 3. The common complaints of unstable particle numbers as well as unpredictable results and speed of branching filters are largely overcome. We also give variants/improvements that use stratified and state-dependent branching. Indeed, it is illustrated in Section 5 that our most basic branching algorithm is significantly faster and more accurate at tracking than all the bootstrap variant algorithms considered here. The outperformance of the branching filter is most extreme when either model selection is required or the tracking problem is difficult. We then study variants of the branching algorithm that give yet better performance or require less computation time.

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

The appendix contains a proof of our stability result, establishing boundedness of particle numbers and weights and Section 4 contains a variance analysis.

2. Resampled Particle Filters

In this section, we review the *resampled particle filters*, starting with the first and most popular *bootstrap algorithm* of Gordon et. al. (1993), but also including: Residual resampling introduced by Liu and Chen (1998), Stratified resampling introduced by Kitagawa (1996), Combined Residual-Stratified resampling discussed in Douc et. al. (2005) and Systematic resampling introduced in Carpenter et. al. (1999). Finally, we explain how to use these algorithms in model selection problems.

2.1. Bootstrap Algorithm.

The *bootstrap particle filter algorithm* is one of the big breakthroughs in big data sequential estimation and its convergence properties have been thoroughly studied in e.g. Del Moral and Miclo (2000). It overcomes the increasing variance weight problem of the weighted filter pointed out in Doucet et. al. (2000). However, it has its limitations in terms of model selection, parallelizability, performance and speed. For clarity, we first summarize the bootstrap algorithm:

Initialize: $\left\{\mathbb{X}_0^k\right\}_{k=1}^N$ are independent initial particle samples of π_0, $V_{N+1}=1$

Repeat: for $n=0,1,2,...$ do

(1) Weight by Observation: $\widehat{\mathbb{L}}_{n+1}^k = \alpha_{n+1}\left(\mathbb{X}_n^k\right)$ for $k=1,2,...,N$

(2) Normalize Weight: $w_{n+1}^k = \frac{\widehat{\mathbb{L}}_{n+1}^k}{\widehat{\mathbb{L}}_{n+1}}$ for $k=1,2,...,N$, where $\widehat{\mathbb{L}}_{n+1} = \sum_{i=1}^N \widehat{\mathbb{L}}_{n+1}^i$

(3) Evolve Independently:

$$P^Y(\widehat{\mathbb{X}}_{n+1}^k \in \Gamma_k \ \forall\ k | \mathcal{F}_n^{\mathbb{X}}) = \prod_{k=1}^N K(\mathbb{X}_n^k, \Gamma_k) \text{ for all } \Gamma_k$$

(4) Estimate π_{n+1} by: $\mathbb{P}_{n+1}^N = \sum_{k=1}^N w_{n+1}^k \delta_{\widehat{\mathbb{X}}_{n+1}^k}$.

(5) Resample: $p_i = \sum_{k=1}^i w_{n+1}^k$ for $i=1,...,N$, $j=N-1$

Repeat: for $k=N,N-1,...,2,1$ do

- Draw $[0,1]$-uniform U_k and set $V_k = U_k^{\frac{1}{k}} V_{k+1}$
- While $V_k \le p_j$ set $j=j-1$
- Set $\mathbb{X}_{n+1}^k \doteq \widehat{\mathbb{X}}_{n+1}^{j+1}$

Remark 1. *We extract our estimate before resampling to avoid excess noise.*

This algorithm is $O(N)$ in operations per particle. In particular, we utilized a clever idea credited to Carpenter et. al. (1999) to keep the resampling to $O(N)$. $(V_1,...,V_N)$ has the joint distribution of the order statistics for $\{U_k\}_{k=1}^N$.

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

The bootstrap algorithm is multinomial resampling since, during resampling, all particles are effectively removed from their locations and then randomly replaced at any of the locations according to the relative weights at the locations. This results in multinomial redistribution of the particles based upon the relative weights. This type of resampling introduces excess noise in the system and tends to degrade performance. It is well understood that other types of resampling that move particles around less and introduce less noise into the particle system are desired.

2.2. **Improved Resampling Methods.** Doucet et. al. (2000) gives an algorithm that alternates between the weighted and bootstrap algorithms depending upon how many *effective particles* there are. This algorithm addresses the tradeoff between introducing resampling noise into the system and coping with continual weight variance increase. However, we argue that it is better for performance to make the resampling decisions on a particle-by-particle basis, which is the focus of this work. Our particle-by-particle approach will also avoid the two separate time problem: a fast time when there is no resampling and a slow one when there is. Some real-time applications are not conducive to sudden switches to slow times. Therefore, we do not consider their algorithm herein but rather limit ourselves to procedures where the resampling is essentially evenly spread out over time. Still, Doucet et. al.'s (2000) *effective number of particles* is a useful concept that will be used in one of our better branching particle algorithms to follow.

There are variations to the bootstrap resampling step (see e.g. Del Moral et. al. (2000); Douc et. al. (2005)) that can be better than the bootstrap algorithm while maintaining relatively even resampling burden. The four following methods have been shown to be unbiased in previous work (see e.g. Douc et. al. (2005)).

2.2.1. *Residual Resampling.* Liu and Chen (1998) introduced residual resampling to reduce resampling noise in the bootstrap filter. The idea is to keep particles at the higher-weight sites (those with weight above the average) so fewer particles are redistributed and less resampling noise is introduced, reducing the number of uniform random variables used from N to R, where R is defined below.

The bootstrap algorithm, given above, is easily modified for this improvement. Instead of step (5) in the bootstrap algorithm, we do the following:

(5) Preserve: $S = 0$

Repeat: for $j = 1, 2, ..., N$ do
- $k = 0$
- While $k < \lfloor N w_{n+1}^j \rfloor$ set $k = k + 1$, $\mathbb{X}_{n+1}^{S+k} \doteq \widehat{\mathbb{X}}_{n+1}^j$
- $S = S + k$

(6) Resample: $R = N - S$; $\overline{p}_i = \sum_{k=1}^{i} \frac{N w_{n+1}^k - \lfloor N w_{n+1}^k \rfloor}{R}$ for $i = 1, ..., N$; $j = N - 1$

Repeat: for $k = N, N-1, ..., 2, S+1$ do
- Draw $[0, 1]$-uniform U_k and set $V_k = U_k^{\frac{1}{k}} V_{k+1}$
- While $V_k \leq \overline{p}_j$ set $j = j - 1$

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

– Set $\mathbb{X}_{n+1}^k \doteq \widehat{\mathbb{X}}_{n+1}^{j+1}$

While the algorithm is a little more complicated than the bootstrap its performance and speed have increased by reducing number of random variables introduced. It is still $O(N)$ but with a smaller constant in front than for the bootstrap algorithm.

2.2.2. *Stratified Resampling.* Kitagawa (1996) introduced a different resampling noise reduction technique. The ideas are to have less randomness in the uniform random variables and avoid the need for ordering the uniforms. The randomness of each uniform random variable is reduced to vary over an interval of length $\frac{1}{N}$.

The bootstrap algorithm, given above, is easily modified for this improvement by replacing Step (5) with:

(5) Resample: $p_i = \sum_{k=1}^{i} w_{n+1}^k$ for $i = 1, ..., N$, $j = 1$

Repeat: for $k = 1, 2, ..., N$ do
– Draw $\left[\frac{k-1}{N}, \frac{k}{N}\right]$-uniform U_k
– While $U_k \geq p_j$ set $j = j + 1$
– Set $\mathbb{X}_{n+1}^k \doteq \widehat{\mathbb{X}}_{n+1}^{j}$

This algorithm is also $O(N)$ with a smaller constant in front than bootstrap. Also, each uniform above has variance $\frac{1}{12N^2}$ versus $\frac{1}{12}$ for the bootstrap. This smaller uniform variance translates into smaller particle system variance in the resampling.

2.2.3. *Systematic Resampling.* Carpenter et. al. (1999) modified the stratified resampling to be more computationally efficient at the cost of giving up conditional independence. Only one uniform random variable is used. Step (5) simply becomes:

(5) Resample: $p_i = \sum_{k=1}^{i} w_{n+1}^k$ for $i = 1, ..., N$, $j = 1$

Draw $\left[-\frac{1}{N}, 0\right]$-uniform U

Repeat: for $k = 1, 2, ..., N$ do
– Set $U_k = U + \frac{k}{N}$
– While $U_k \geq p_j$ set $j = j + 1$
– Set $\mathbb{X}_{n+1}^k \doteq \widehat{\mathbb{X}}_{n+1}^{j}$

One must be warned that there are examples where the systematic resampling scheme behaves very poorly (see Douc et. al. (2005)). Indeed, it is not one of the better performers for our problems considered in Section 5.

2.2.4. *Combined Resampling.* Since the Residual and Stratified resampling methods reduce the randomness in different ways, the combination might produce a yet better method (see Douc et. al. (2005)). Step (5) in the bootstrap is replaced with:

(5) Preserve: $S = 0$

Repeat: for $j = 1, 2, ..., N$ do
– $k = 0$

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

– While $k < \lfloor N w_{n+1}^j \rfloor$ set $k = k+1$, $\mathbb{X}_{n+1}^{S+k} \doteq \widehat{\mathbb{X}}_{n+1}^j$
– $S = S + k$

(6) Resample: $R = N - S$; $\overline{p}_i = \sum_{k=1}^{i} \frac{N w_{n+1}^k - \lfloor N w_{n+1}^k \rfloor}{R}$ for $i = 1, ..., N$; $j = 1$

Repeat: for $k = 1, 2, ..., R$ do

– Draw $\left[\frac{k-1}{R}, \frac{k}{R}\right]$-uniform U_k
– While $U_k \geq \overline{p}_j$ set $j = j + 1$
– Set $\mathbb{X}_{n+1}^{S+k} \doteq \widehat{\mathbb{X}}_{n+1}^j$

This algorithm is also $O(N)$ with a smaller constant than bootstrap. Combined Resampling clearly improves Residual Resampling but it is unclear if it improves Stratified. Stratified uses N random variables with variance $\frac{1}{12N^2}$ while Combined resampling uses R random variables with variance $\frac{1}{12\,R^2}$. Which produces less resampling noise is not obvious without further analysis.

2.3. **Model Selection.** In most real tracking problems, one never knows which is the best mathematical model of reality so it makes sense to let the data decide. This leads us to the combined problem of model selection and tracking. As previously mentioned, the unnormalized filter total mass $\sigma_n(1)$ gives Bayes factor of the observations containing a given signal model (i.e. $Y_j = h(X_{j-1}) + V_j$ for $j \leq n$) to pure noise (i.e. $Y_j = V_j$ for $j \leq n$). The ratio of two unnormalized filters $B_n^{12} = \frac{\sigma_n^{(1)}(1)}{\sigma_n^{(2)}(1)}$ for two different models ($Y_j = h(X_{j-1}^1) + V_j$ for $j \leq n$ and $Y_j = h(X_{j-1}^2) + V_j$ for $j \leq n$) then gives of Bayes factor for signal 1 versus signal 2.

Del Moral and Miclo (2000, p. 16) show us how to recover the unnormalized filter and thereby do model selection using bootstrap-type algorithms. By Bayes rule and (1.2), one finds that

$$\pi_{n-1}(\alpha_n) = \frac{\sigma_{n-1}(\alpha_n)}{\sigma_{n-1}(1)} = \frac{\sigma_n(1)}{\sigma_{n-1}(1)} \Rightarrow \sigma_n(1) = \prod_{m=1}^{n} \pi_{m-1}(\alpha_m), \tag{2.1}$$

where α_m is defined in (1.1). Hence, to do model selection in the bootstrap-type algorithms, one can just add the following after step (1):

(1a) Model Selection: $\sigma_{n+1}(1) = \sigma_n(1)\frac{\widehat{\mathbb{L}}_n}{N}$, where $\widehat{\mathbb{L}}_n = \sum_{i=1}^{N} \widehat{\mathbb{L}}_n^i$.

We then need not calculate $\widehat{\mathbb{L}}_n$ in step (2) but must add $\sigma_0(1) = 1$ to the initialize.

3. Branching Particle Filters

In this section, we introduce a class of branching particle filters. We no longer necessarily have full resampling but rather allow partial resampling and weight propagation. In one extreme case, we have the weighted particle filter where no resampling takes place and weights are always propagated. In the other extreme case, we have a fully resampled particle filter, which can be thought of as a branching

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

alternative to the residual resampling or combined resampling particle filter. In between, we have a whole class of branching particle filters with a flexible amount of resampling that affords an effective tradeoff between weight variance increase (the weighted particle issue) and resampling noise (the bootstrap particle issue). We first describe the branching particle filters in terms of uniform random variables $\{\mathbb{U}_n^k\}$ used to create the branching variables $\{\rho_n^k\}$ in two different ways.

The following branching Markov process $\{\mathbb{S}_n^N, n = 0, 1, ...\}$ approximates the *unnormalized* filter $\{\sigma_n, n = 0, 1, ...\}$ in terms of the observations as follows:

Initialize: $\left\{\mathbb{X}_0^k\right\}_{k=1}^N$ are independent samples of π_0, $\mathbb{N}_0 = N$, $\mathbb{N}_n = 0$ for all $n \in \mathbb{N}$ and $\mathbb{L}_0^k = 1$ for $k = 1, ..., N$.

Repeat: for $n = 0, 1, 2, ...$ do

(1) Weight by Observation: $\widehat{\mathbb{L}}_{n+1}^k = \alpha_{n+1}\left(\mathbb{X}_n^k\right)\mathbb{L}_n^k$ for $k = 1, 2, ..., \mathbb{N}_n$
(2) Evolve Independently:

$$P^Y(\widehat{\mathbb{X}}_{n+1}^k \in \Gamma_k \ \forall\ k | \mathcal{F}_n^{\mathbb{X}} \vee \mathcal{F}_{n+1}^{\mathbb{U}}) = \prod_{k=1}^{\mathbb{N}_n} K(\mathbb{X}_n^k, \Gamma_k) \quad \forall \Gamma_k$$

(3) Estimate σ_{n+1} by: $\mathbb{S}_{n+1}^N = \dfrac{1}{N}\displaystyle\sum_{k=1}^{\mathbb{N}_n} \widehat{\mathbb{L}}_{n+1}^k \delta_{\widehat{\mathbb{X}}_{n+1}^k}$ and $\pi_{n+1}(f)$ by $\frac{\mathbb{S}_{n+1}^N(f)}{\mathbb{S}_{n+1}^N(1)}$.
(4) Average Weight: $\mathbb{A}_{n+1} = \mathbb{S}_{n+1}^N(1)$

Repeat (5-6): for $k = 1, 2, ..., \mathbb{N}_n$ do

(5) Resampled Case: If $\widehat{\mathbb{L}}_{n+1}^k \notin (a_n\mathbb{A}_{n+1}, b_n\mathbb{A}_{n+1})$ then
  (a) Offspring Number: $\mathbb{N}_{n+1}^k = \left\lfloor \frac{\widehat{\mathbb{L}}_{n+1}^k}{\mathbb{A}_{n+1}} \right\rfloor + \rho_{n+1}^k$, with ρ_{n+1}^k a $\left(\frac{\widehat{\mathbb{L}}_{n+1}^k}{\mathbb{A}_{n+1}} - \left\lfloor \frac{\widehat{\mathbb{L}}_{n+1}^k}{\mathbb{A}_{n+1}} \right\rfloor\right)$-Bernoulli
  (b) Resample: $\mathbb{L}_{n+1}^{\mathbb{N}_{n+1}+j} = \mathbb{A}_{n+1}, \mathbb{X}_{n+1}^{\mathbb{N}_{n+1}+j} = \widehat{\mathbb{X}}_{n+1}^k$ for $j = 1, ..., \mathbb{N}_{n+1}^k$
  (c) Add Offspring Number: $\mathbb{N}_{n+1} = \mathbb{N}_{n+1} + \mathbb{N}_{n+1}^k$
(6) Non-resample Case: If $\widehat{\mathbb{L}}_{n+1}^k \in (a_n\mathbb{A}_{n+1}, b_n\mathbb{A}_{n+1})$ then
  $\mathbb{N}_{n+1} = \mathbb{N}_{n+1} + 1$, $\mathbb{L}_{n+1}^{\mathbb{N}_{n+1}} = \widehat{\mathbb{L}}_{n+1}^k$, $\mathbb{X}_{n+1}^{\mathbb{N}_{n+1}} = \widehat{\mathbb{X}}_{n+1}^k$

Remark 2. *We extract our estimate before resampling to avoid excess noise. Key steps (5,6) determine the new number of particles $\mathbb{N}_{n+1}$ and weights $\mathbb{L}_{n+1}^k$ in an unbiased manner. When the prior weight $\widehat{\mathbb{L}}_{n+1}^k$ for particle k is extreme we do residual-style branching, splitting particles as deterministically as possible in (5). The result is zero or more particles all having the average weight at the same location as the parent. When the prior weight $\widehat{\mathbb{L}}_{n+1}^k$ is not extreme we run a weighted particle step in (6). The flexibility in this class of algorithms is in how we determine "extreme".*

Remark 3. *$\mathbb{A}_{n+1}$, $\mathbb{L}_{n+1}^k$ and $\widehat{\mathbb{L}}_{n+1}^k$ actually depend upon the initial number of particles N. Occasionally, we will stress this fact by relabeling $\mathbb{A}_{n+1}$ as $\mathbb{A}_{n+1}^N$.*

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

We are not the first to use branching particle filters for tracking. Indeed, we were inspired by Crisan and Lyons (1997) and Ballantyne et. al. (2000). However, our algorithms differ from the ones in those papers and our goals are also different.

After establishing the appropriate bounds on $\mathbb{N}_{n+1}$ in Theorem 2 to follow, we can easily see that this algorithm is also $O(N)$. Indeed, a careful comparison of this algorithm to the prior ones leads us to the believe that the constant implied in the $O(N)$ notation for the branching algorithm may be smaller than that for the bootstrap, especially when the Resampled Case does not occur too often. We will establish this fact experimentally below. Since σ_n is estimated both model selection and filtering can be done simultaneously without adding the additional step.

3.1. **Residual Branching Filter.** Like resampled filters, there are various ways to introduce the randomness, in this case the $\{\rho_{n+1}^k\}_{k=1}^{\mathbb{N}_n}$ for unbiased branching. The choice affects performance and implementation ease. A simple possibility is:

i) Let $\{\mathbb{U}_{n+1}^k\}_{k=1}^{\mathbb{N}_n}$ be independent $[0,1]$-Uniform RVs.
ii) Set $\rho_{n+1}^k = 1_{\mathbb{U}_{n+1}^k \le \left(\frac{\widehat{\mathbb{L}}_{n+1}^k}{\mathbb{A}_{n+1}} - \left\lfloor \frac{\widehat{\mathbb{L}}_{n+1}^k}{\mathbb{A}_{n+1}} \right\rfloor\right)}$.

In this way, the $\{\rho_{n+1}^k\}_{k=1}^{\mathbb{N}_n}$ are independent of each other and everything else. The reason for the *Residual* Branching labeling is, similar to Residual Resampling, we first create as many particles as we can deterministically and then we allocate the remaining offspring using independent uniform random variables.

3.2. **Combined Branching Filter.** We can add stratified resampling to help control the number of particles and improve performance. When $a_n \approx b_n$, we:

i) Let $\{\mathbb{V}_{n+1}^k\}_{k=1}^{\mathbb{N}_n}$ be independent with $\mathbb{V}_{n+1}^k \sim \left[\frac{k-1}{\mathbb{N}_n}, \frac{k}{\mathbb{N}_n}\right]$-Uniform and $\mathbb{U}_{n+1}^k = \mathbb{V}_{n+1}^{p(k)}$, where p is a random permutation of $\{1, 2, ..., N_n\}$ uniformly distributed over the set of all permutations.
ii) Set $\rho_{n+1}^k = 1_{\mathbb{U}_{n+1}^k \le \left(\frac{\widehat{\mathbb{L}}_{n+1}^k}{\mathbb{A}_{n+1}} - \left\lfloor \frac{\widehat{\mathbb{L}}_{n+1}^k}{\mathbb{A}_{n+1}} \right\rfloor\right)}$.

Then, the $\{\rho_{n+1}^k\}_{k=1}^{\mathbb{N}_n}$ are exchangeable but not independent of each other. (They are actually negatively correlated, which is desireable for particle control.) The advantage of this approach is it is not possible to get mostly large or mostly small uniform random numbers so the number of particles will vary less.

In the usual case, where $a_n \ll b_n$ so many particles are not resampled, there is a better stratified method. We replace (5-6) in the basic branching algorithm with:

5) Non-resample count: $l = 0$
6) For $k = 1, 2, ..., \mathbb{N}_n$ do
   If $\widehat{\mathbb{L}}_{n+1}^k \notin (a_n\mathbb{A}_{n+1}, b_n\mathbb{A}_{n+1})$ then: $\widehat{\mathbb{L}}_{n+1}^{k-l} = \widehat{\mathbb{L}}_{n+1}^k, \widehat{\mathbb{X}}_{n+1}^{k-l} = \widehat{\mathbb{X}}_{n+1}^k$
   Otherwise: $l = l + 1$, $\mathbb{L}_{n+1}^l = \widehat{\mathbb{L}}_{n+1}^k$, $\mathbb{X}_{n+1}^l = \widehat{\mathbb{X}}_{n+1}^k$
7) Let $\mathbb{N}_{n+1} = l$, $\{\mathbb{V}_{n+1}^k\}_{k=l+1}^{\mathbb{N}_n}$ be independent with $\mathbb{V}_{n+1}^k \sim \left[\frac{k-1}{\mathbb{N}_n - l}, \frac{k}{\mathbb{N}_n - l}\right]$-Uniform, p be a random permutation of $\{l+1, l+2, ..., \mathbb{N}_n\}$, $\mathbb{U}_{n+1}^k = \mathbb{V}_{n+1}^{p(k)}$

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

8) For $k = l+1, l+2, ..., \mathbb{N}_n$ do
$\mathbb{N}_{n+1}^k = \left\lfloor \frac{\widehat{\mathbb{L}}_{n+1}^{k-l}}{\mathbb{A}_{n+1}} \right\rfloor + 1_{\mathbb{U}_{n+1}^k \leq \left(\frac{\widehat{\mathbb{L}}_{n+1}^{k-l}}{\mathbb{A}_{n+1}} - \left\lfloor \frac{\widehat{\mathbb{L}}_{n+1}^{k-l}}{\mathbb{A}_{n+1}} \right\rfloor \right)}$
$\mathbb{L}_{n+1}^{\mathbb{N}_{n+1}+j} = \mathbb{A}_{n+1}, \mathbb{X}_{n+1}^{\mathbb{N}_{n+1}+j} = \widehat{\mathbb{X}}_{n+1}^{k-l}$ for $j = 1, ..., \mathbb{N}_{n+1}^k$
$\mathbb{N}_{n+1} = \mathbb{N}_{n+1} + \mathbb{N}_{n+1}^k$

3.3. **Resampling Control.** The amount of and conditions for branching is controlled by the parameters a_n, b_n. In one extreme case, we could choose $a_n = 0$ and $b_n = \infty$ and find that no resampling takes place. We then have the weighted particle filter. In the other extreme case, we always resample if $a_n = b_n$ for all n. This case is the closest to the resampled particle filters considered above.

Generally, a_n and b_n should have geometric center of 1 to keep the number of particles fairly constant. Suppose otherwise that $a_n = 0$ and $b_n = 1$. Then, we only branch the below-average weight particles and expect fewer particles after branching.

There is some inherent particle control if a_n and b_n to have geometric center of 1 and this is due to the fact that $\mathbb{A}_{n+1}$ is normalized by N and not $\mathbb{N}_n$. To see this, we consider the simplest case where $a_n = b_n = 1$, which corresponds to complete branching. Then, examining the branching algorithm, we find (by Step (5a)) that the total expected number of particles after resampling is:

$$\sum_{j=1}^{\mathbb{N}_n} E[\mathbb{N}_{n+1}^j \big| \mathcal{F}_{n+1}^{\widehat{\mathbb{L}}} \vee \mathcal{F}_n^{\widehat{\mathbb{N}}}] = \sum_{j=1}^{\mathbb{N}_n} \frac{N\widehat{\mathbb{L}}_{n+1}^j}{\sum_{k=1}^{\mathbb{N}_n} \widehat{\mathbb{L}}_{n+1}^k} = N \tag{3.1}$$

regardless of whether residual or combined branching is used. This particle control becomes yet more pronounced when combined with the negatively correlated ρ's produced by the stratified scheme within the combined branching particle algorithm. More unbiased particle control could be added. However, we are yet to see an example where it is warranted (see the results below).

Since weights multiply and start at 1, we generically take $a_n = \frac{1}{r_n}$ and $b_n = r_n$ for some $r_n \in [1, \infty]$. Moreover, the new multiplicative weight update has the form $\alpha_n(X_{n-1}) = \frac{g(Y_n - h(X_{n-1}))}{g(Y_n)}$, which adapts for noisy observations. However, it does not account well for only having partial measurements of the signal. As a simple example, if we have range (alternatively bearing) only measurements of a position-velocity model in the plane, then one observation has no information about velocity and only partial measurement of position. If complete resampling were used, then particles would (with high probability) accumulate on an arc (in a line of sight) with matching velocities and other positional component. If these velocities are all wrong (which can easily happen with a finite number of particles), then the majority of the particles will head in the wrong direction. (A similar issue occurs if incorrect bearings of a few particles matching the observed range are copied to all or most.) To avoid these types of situations, one should generically choose a larger r_n when it would take several observations to get a good idea about whole signal. In all our experiments to date, we are yet to see a problem where either no resampling or full resampling (like the resampled particle filters) preforms best.

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

In the experiments below, we will first consider the case where $r_n = r$ does not depend upon n. However, we can have adaptive resampling when r_n depends upon the branching particle system. One example of this is *dynamic branching*, where

$$r_n = \exp\left(c \left[\frac{1}{N_n} \sum_{k=1}^{N_n} (\ln \widehat{\mathbb{L}}_{n+1}^k)^2 - \left(\frac{1}{N_n} \sum_{k=1}^{N_n} \ln \widehat{\mathbb{L}}_{n+1}^k \right)^2 \right]^{\frac{q}{2}} \right),$$

with $c, q > 0$. A larger $q > 1$ (smaller $q < 1$), with c adjusted to maintain the same average amount of branching, would be used in the situation where one wanted more resampling when system entropy low (high). To explain, we imagine $\xi^k = \ln \widehat{\mathbb{L}}_{n+1}^k - \ln \mathbb{A}_{n+1}$ are independent, zero-mean and Gaussian. Then, r_n with $q = 1$ would correspond to a relatively fixed (67% when $c = 1$) number of particles being resampled regardless of disorder. Taking $q < 1$ and compensating with $c > 1$ to maintain the same average amount of resampling would then cause more resampling at times when there is more entropy. This is investigated further experimentally below.

Another, more direct, way to handle uneven weights is through the *effective* number of particles estimate, N^{eff}, as discussed in Doucet et. al. (2000). In our setting, the effective and non-effective particle estimates are:

$$\mathbb{N}_{n+1}^{eff} = \frac{N^2 \mathbb{A}_{n+1}^2}{\sum_{k=1}^{\mathbb{N}_n} \left(\widehat{\mathbb{L}}_{n+1} \right)^2}, \quad \mathbb{N}_{n+1}^{noneff} = \mathbb{N}_{n+1} - \mathbb{N}_{n+1}^{eff}.$$

It very reasonable to anticipate better results when branching either more or fewer particles in the situation there are few effective ones. A first intuition might lead us to the conclusion that it is better to branch more in order to obtain more effective particles immediately. However, reviewing the range (and bearing) only example mentioned previously also leads us to the realization that this too could be wrong. We do not assume either a priori but rather in *effective particle branching* set

$$r_n = \frac{c^{eff} \mathbb{N}_{n+1}^{eff} + c^{noneff} \mathbb{N}_{n+1}^{noneff}}{\mathbb{N}_{n+1}} = c^{noneff} + (c^{eff} - c^{noneff}) \frac{\mathbb{N}_{n+1}^{eff}}{\mathbb{N}_{n+1}} \tag{3.2}$$

for experimentally determined constants c^{eff}, $c^{noneff} > 0$ and let the data decide.

3.4. **Stability and Particle Control.** Most authors have rejected branching particle filters due to possible instability of the number of particles as well as the computational consequences of this instability. This rejection was too hasty. Yes, the algorithm can fail. During resampling, there is a possibility of immediately killing all particles if $\max_{j \le \mathbb{N}_{n-1}} \frac{N \widehat{\mathbb{L}}_n^j}{\sum_{i=1}^{\mathbb{N}_n} \widehat{\mathbb{L}}_n^i} < 1$. Ironically, this can only happen if there are more particles than at start. However, it may still be possible to degenerate immediately to one particle when $\mathbb{N}_n \le N$. Conversely, it is not possible to increase by more than $N - 1$ particles in one step. Weight variation is also a big concern: $\mathbb{L}_n^j$ can become very uneven as N increases. Some regularity results are required to

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

ensure that there are enough effective particles and moment bounds to justify the anticipation of rate of convergence results as $N \to \infty$.

The following *one step bounds* ensure the risk of such system irregularity decreases exponentially in the initial number of particles, a first step in *disproving* the opinion *branching particle filters are unstable in the number of particles*. Let $B(\mathbb{R}^d)$ be the bounded and $C_{++}(\mathbb{R}^d)$ be the strictly-positive continuous functions on $\mathbb{R}^d$.

Theorem 2. *Suppose residual branching; $h \in B(\mathbb{R}^d)$; and $g \in C_{++}(\mathbb{R}^d)$. Then, there are $\epsilon_n > 0$, $C_n > 1$ and $\mathbb{D}_n^N \in \sigma\{\mathbb{N}_l,\ l \leq n\}$ such that $\mathbb{D}_{n+1}^N \subset \mathbb{D}_n^N$ for all $n = 0, 1, 2...$; $Q^Y\left(\mathbb{D}_n^N\right) \geq 1 - 2ne^{-\epsilon_n N}$ for $N \geq 1$; and*

$$\frac{1}{C_n} \leq \frac{\mathbb{N}_l}{N}, \mathbb{L}_l^i, \mathbb{A}_l^N \leq C_n \ \forall \in \{1, ..., \mathbb{N}_l\}, l \in \{0, ..., n\} \text{ on } \mathbb{D}_{n-1}^N.$$

Remark 4. *This result is proved in the appendix. The result bounds the number of particles and also says that the algorithm is well behaved for at least one step on the large exchangeable set $\mathbb{D}_n^N$. This result is for residual branching. However, adding stratification as in the combined branching only adds particle control through the negatively correlated ρ's.*

We now look at particle variation experimentally. The expected number of particles is always the initial number N by (3.1). There still could be wide variation, especially for small number of particles, but the following graphs show that this is not the case. The smallest number of initial particles that we looked at was 500 and it was in the Range Only model (defined in Section 5). In this case, the particle number standard deviation of Residual Branching was 120 or 24%. Moreover, the Combined Branching algorithm showed a significant improvement to 90 or 18%.

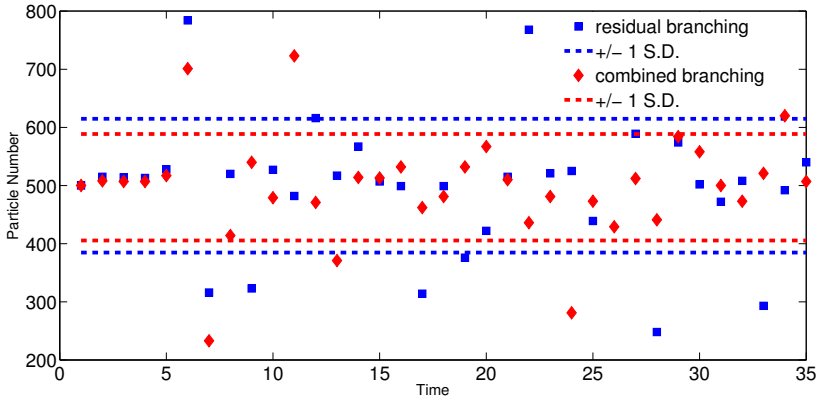

FIGURE 1. Particle Number of *Range Only Model* with N=500

While these results are random, they seem typical. Certainly, the results to follow on speed and performance are supportive of small particle variations.

We then looked at $N = 2000$ initial particles for the Test Model (see Section 5). The particle number standard deviations of the Residual Branching and Combined branching were then 165 and 130 or 8.25% and 6.5%.

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

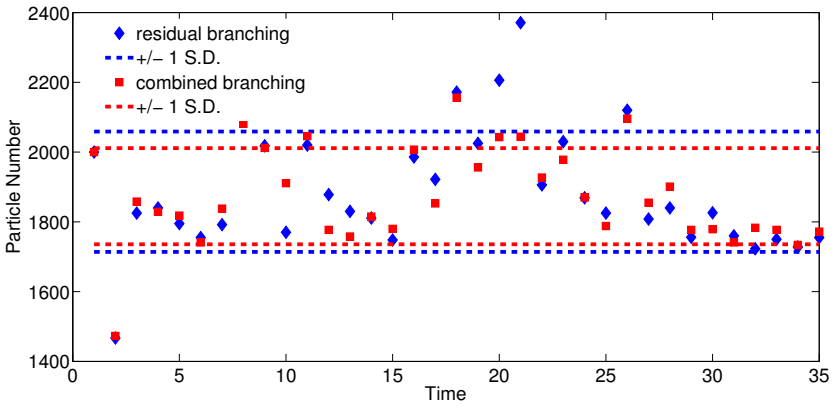

FIGURE 2. Particle Number of *Test Model* with N=2000

Finally, we looked at $N = 10,000$ intial particles for both models. The particle number standard deviations of the Residual Branching and Combined branching on the Test model were then 295 and 45 or 2.95% and 0.45%.

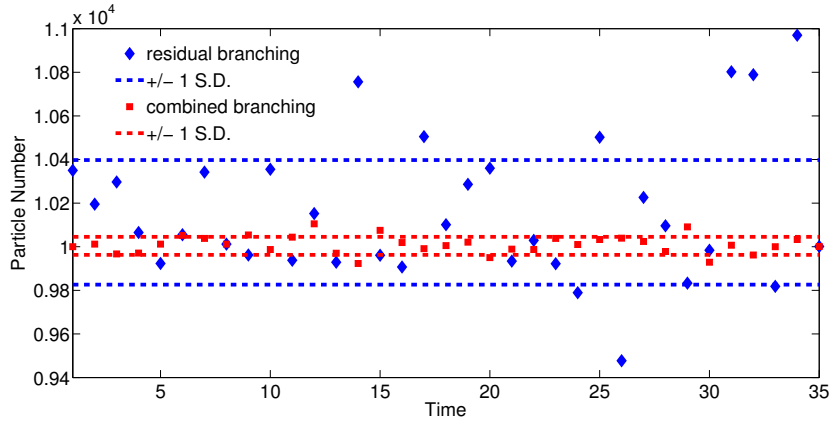

FIGURE 3. Particle Number of *Test Model* with N=10,000

The particle number standard deviations of the Residual Branching and Combined branching on the Range Only model were then 415 and 320 or 4.15% and 3.2%.

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

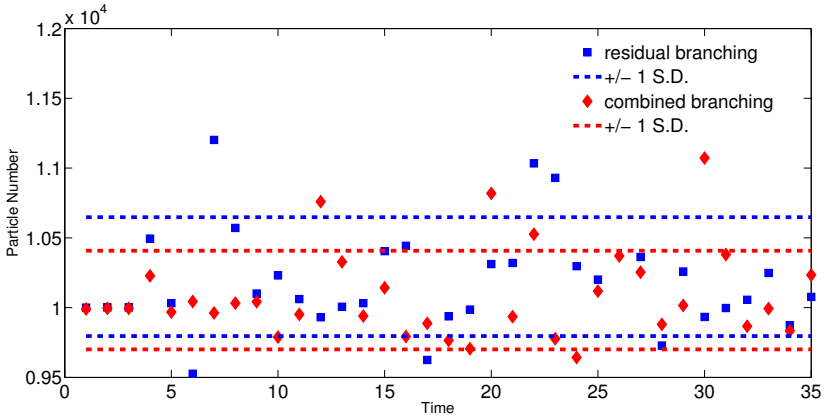

FIGURE 4. Particle Number of *Range Only Model* with N=10,000

We can make the following experimental conclusions:

(1) The particle numbers did not vary much, provided enough particles were used to be able to estimate the signal reasonably. Indeed, we will see below that the particle variations did not adversely affect error nor execution speed.
(2) Combined Branching did a better job of keeping the number of particles constant than the basic Residual Branching.
(3) The relative variation of the number of particles generally decreased in the number of initial particles. It is speculated that some law of large numbers effect takes place.

Further, it is clear that the particles numbers varied less for the Test model than for the Range Only. It was felt that this was due to the fact that simpler problems have less variations throughout and are solved well with fewer particles.

4. Variance Analysis

It is difficult to compare the resampled and branching particle filters since they approximate different things, the regular and the unnormalized filters. Moreover, many particle filters are hard to analyze due to the lack of independence. However, it is relatively easy to consider the noise introduced by the first resampling or branching through a look at the expected conditional variance at that time. Prior to this time all systems, whether resampling or branching, will have the same particles (locations and weights). Hence, it is the spot where the first difference occurs.

Even restricting ourselves to this first resampling or branching event, we can not assume independence since the stratified variables are not independent. Instead, we recall:

Definition 1. *Random variables* $\{\xi_1, \xi_2, ..., \xi_N\}$ *are exchangeable if* $\{\xi_1, \xi_2, ..., \xi_N\} \stackrel{\mathcal{D}}{=} \{\xi_{\pi(1)}, \xi_{\pi(2)}, ..., \xi_{\pi(N)}\}$ *for any permutation* π *of* $\{1, 2, ..., N\}$.

I.i.d. random variables are exchangeable. Also, random permuted collections are clearly exchangeable. Suppose $\{\xi_1, \xi_2, ..., \xi_N\}$ are exchangeable random variables on

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

$(\Omega, \mathcal{F}, P)$ and $\mathcal{G} \subset \mathcal{F}$ is a σ-algebra. Then, the expected conditional variance is

$$\begin{aligned}\gamma &= N^{-2} E\left\{E\left[(\xi_1+\xi_2+\cdots+\xi_N)^2\middle|\mathcal{G}\right] - \left(E\left[\xi_1+\xi_2+\cdots+\xi_N\middle|\mathcal{G}\right]\right)^2\right\} \qquad (4.1)\\ &= N^{-2}\left\{\sum_{i=1}^{N} E\left[\xi_i^2\right] + \sum_{\substack{i,j=1\\ i\neq j}}^{N} E\left[\xi_i\xi_j\right] - \sum_{i,j=1}^{N} E\left(E\left[\xi_i\middle|\mathcal{G}\right] E\left[\xi_j\middle|\mathcal{G}\right]\right)\right\}\\ &= \frac{E[\xi_1^2]}{N} + \frac{N-1}{N} E\left[\xi_1\xi_2\right] - N^{-2}\sum_{i,j=1}^{N} E\left(E\left[\xi_i\middle|\mathcal{G}\right] E\left[\xi_j\middle|\mathcal{G}\right]\right).\end{aligned}$$

When applied to particle filtering, $\mathcal{G}$ will be the information in all particles and weights prior to the resample or branch. In our first-event consideration, $\mathcal{G} = \sigma\{(\widehat{\mathbb{L}}_1^k, \widehat{\mathbb{X}}_1^k),\ k = 1, 2, ..., N\}$ and our unbiased construction means

$$E[\xi_k|\mathcal{G}] = 0$$

so the expected conditional variance formula (4.1) becomes

$$\gamma = \frac{E[\xi_1^2]}{N} + \frac{N-1}{N} E\left[\xi_1\xi_2\right]. \qquad (4.2)$$

Both terms will depend upon the precise type of resampling or branching.

In stratified resampling, combined resampling and combined (residual-stratified) branching, we replace independence in our uniform random variables with desired negative correlations. Indeed, the joint distribution of the randomly-permuted stratified uniforms $\mathbb{U}_1^1, \mathbb{U}_1^2$ becomes:

$$\begin{aligned}& P(\mathbb{U}_1^1 \in [c_1, d_1], \mathbb{U}_1^2 \in [c_2, d_2]) \qquad (4.3)\\ = \ & \frac{R^2}{R(R-1)} P(V \in [c_1, d_1]) P(V \in [c_2, d_2])\\ - \ & \frac{1}{R(R-1)} \sum_{j=0}^{R-1} P\left(\frac{V+j}{R} \in [c_1, d_1]\right) P\left(\frac{V+j}{R} \in [c_2, d_2]\right),\end{aligned}$$

where V is a $[0,1]$-uniform random variable and R ($\leq \mathbb{N}_n$) is the (possibly random) number of uniform random variables used (which should be at least as many as the number of particles that are resampled/branched). Intuitively, the formula says that the joint (condtional) distribution of $\mathbb{U}_1^1, \mathbb{U}_1^2$ is the distribution of two independent $[0,1]$-uniform random variables scaled up by $\frac{R^2}{R(R-1)}$ minus the sum of scaled down distributions of two independent $[\frac{j}{R}, \frac{j+1}{R}]$-uniform random variables. This implies the following expectation formula:

$$\begin{aligned}E[f(\mathbb{U}_1^1) g(\mathbb{U}_1^2)] \ &= \ \frac{R^2}{R(R-1)} \int_0^1 \int_0^1 f(u^1) g(u^2) du^1 du^2 \qquad (4.4)\\ &- \ \frac{1}{R(R-1)} \sum_{j=0}^{R-1} \int_j^{j+1} \int_j^{j+1} f\left(\frac{u^1}{R}\right) g\left(\frac{u^2}{R}\right) du^1 du^2.\end{aligned}$$

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

For ease of comparison, we introduce

$$\ell_1^k = \frac{\widehat{\mathbb{L}}_1^k}{\mathbb{A}_1} = N w_1^k, \tag{4.5}$$

where $\mathbb{A}_1 = \frac{1}{N}\sum_{i=1}^N \widehat{\mathbb{L}}_1^i$ for any particle filter. It is important to remember that while ℓ_1^k is a relative weight for the k^{th} particle it still depends upon the other particles. Finally, we emphasize that this analysis is done with the real world probability P.

4.1. **Resampled Filters.** Some conditional variance analysis is done in other works. However, we redo it in our notation for ease of comparison. Recall from the residual and combined resampled algorithms given previously that $R = N - \sum_{j=1}^N \lfloor l_1^j \rfloor$ is the number of particles resampled when residual (or combined) resampled methods are used. The change in the filter due to the first resample is

$$\frac{1}{N}\sum_{j=1}^N \xi_j, \text{ where } \xi_j = \begin{cases} \left(\lfloor \ell_1^j \rfloor + \rho_1^j - \ell_1^j\right) f(\widehat{\mathbb{X}}_1^j) & \text{if residual, combined} \\ \left(\rho_1^j - \ell_1^j\right) f(\widehat{\mathbb{X}}_1^j) & \text{if bootstrap, stratified} \end{cases}. \tag{4.6}$$

Moreover,

$$\rho_1^j = \sum_{k=1}^R 1_{\mathbb{U}_k \in [\overline{p}_{j-1}, \overline{p}_j)}, \quad \text{where} \tag{4.7}$$

Method	R	$\overline{p}_j$	$\mathbb{U}_k$	$\mathbb{U}_k$-character
Bootstrap	N	$\sum_{i\le j} \frac{\ell_1^i}{N}$	U_k	i.i.d.
Residual	R	$\sum_{i\le j} \frac{\ell_1^i - \lfloor \ell_1^i \rfloor}{R}$	U_k	i.i.d.
Stratified	N	$\sum_{i\le j} \frac{\ell_1^i}{N}$	$\pi(U_k)$	Negatively correlated
Combined	R	$\sum_{i\le j} \frac{\ell_1^i - \lfloor \ell_1^i \rfloor}{R}$	$\pi(U_k)$	Negatively correlated

Here, $\mathbb{U}_k$ are $[0,1]$-uniform and π is a random permutation. Note that we did not include this permutation in the stratified and combined methods. However, since the particles and weights ℓ_1^i are exchangeable, the joint distribution of the ρ's is unchanged with or without this permutation.

4.1.1. *Residual Resampling.* $R = 0$ is trivial so consider $R \in \mathbb{N}$. Here, $\{\rho_1^j\}_{j=1}^N$ is (conditionally on $\mathcal{G}$) $\left(R, \frac{\ell_1^1 - \lfloor \ell_1^1 \rfloor}{R}, ..., \frac{\ell_1^N - \lfloor \ell_1^N \rfloor}{R}\right)$-multinomial. Hence, we find that

$$E[\xi_1^2] = E\left[\left\{\ell_1^1 - \lfloor \ell_1^1 \rfloor - \frac{\left(\ell_1^1 - \lfloor \ell_1^1 \rfloor\right)^2}{R}\right\} f^2(\widehat{\mathbb{X}}_1^1)\right] \tag{4.8}$$

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

and by (4.6,4.7) that

$$E[\xi_1\xi_2] = E\left[\left\{\rho_1^1\rho_1^2 - \left(\ell_1^1 - \lfloor \ell_1^1 \rfloor\right)\left(\ell_1^2 - \lfloor \ell_1^2 \rfloor\right)\right\} f(\widehat{\mathbb{X}}_1^1) f(\widehat{\mathbb{X}}_1^2)\right] \tag{4.9}$$
$$= -E\left[\frac{1}{R}\left(\ell_1^1 - \lfloor \ell_1^1 \rfloor\right)\left(\ell_1^2 - \lfloor \ell_1^2 \rfloor\right) f(\widehat{\mathbb{X}}_1^1) f(\widehat{\mathbb{X}}_1^2)\right].$$

Combining these with (4.2), we find that

$$\gamma_1^{RR} = E\left[\left\{\ell_1^1 - \lfloor \ell_1^1 \rfloor - \frac{\left(\ell_1^1 - \lfloor \ell_1^1 \rfloor\right)^2}{R}\right\}\frac{f^2(\widehat{\mathbb{X}}_1^1)}{N}\right] \tag{4.10}$$
$$- \frac{N-1}{N} E\left[\frac{1}{R}\left(\ell_1^1 - \lfloor \ell_1^1 \rfloor\right)\left(\ell_1^2 - \lfloor \ell_1^2 \rfloor\right) f(\widehat{\mathbb{X}}_1^1) f(\widehat{\mathbb{X}}_1^2)\right].$$

Example 1. *(4.10) could appear to produce (impossible) negative values with fixed weights. However, this is not a real possibility when exchangeability is taken into account. For simplicity, suppose* $l_1^1 = 1.99$, $l_1^2 = 0.01$, $l_1^3 = 1$ *and* $f \equiv 1$ *so* γ_1^{RR} *should be* 0. *Then,* $R = 1$ *and (4.10) becomes:*

$$\gamma_1^{RR} = \frac{0.99 \times 0.01}{3} - \frac{2}{3}(0.99)(0.01) < 0, \tag{4.11}$$

which is impossible. However, these fixed l *are also not exchangeable. Instead suppose* l_1^1, l_1^2 *and* l_1^3 *took on these three values and each of the six combinations was equally likely. Then, (4.10) becomes:*

$$\gamma_1^{RR} = \frac{1}{3}\left\{\frac{0.01 - 0.01^2}{3} + 0 + \frac{0.99 - 0.99^2}{3}\right\} - \frac{2}{3}\frac{1}{3}\left\{0 + 0.0099 + 0\right\} = 0, \tag{4.12}$$

as expected. Note: we could bypass the final simplication in (4.1) and allow more general ℓ*'s. However, more complicated formulae would result.*

4.1.2. *Bootstrap Filter.* Following the arguments in the prior Residual Resampling Subsection, we find that

$$\gamma_1^{BS} = E\left[\left\{\ell_1^1 - \frac{\left(\ell_1^1\right)^2}{N}\right\}\frac{f^2(\widehat{\mathbb{X}}_1^1)}{N}\right] - \frac{N-1}{N^2} E\left[\ell_1^1\ell_1^2 f(\widehat{\mathbb{X}}_1^1) f(\widehat{\mathbb{X}}_1^2)\right].$$

4.1.3. *Combined Resampling.* Take $R \neq 0$ again. The $\{\rho_1^j\}_{j=1}^R$ are not multinomial so we have to work with the $\mathbb{U}_i$ and use (4.4). Consequently, it follows that

$$E[\xi_1^2] = E\left[\{(\rho_1^1)^2 - (\ell_1^1 - \lfloor \ell_1^1 \rfloor)^2\} f^2(\widehat{\mathbb{X}}_1^1)\right] \tag{4.13}$$
$$= E\left[\left\{\sum_{k=1}^R 1_{\mathbb{U}_k \in [0,\overline{p}_1)} + \sum_{i \neq k} 1_{\mathbb{U}_k \in [0,\overline{p}_1)} 1_{\mathbb{U}_i \in [0,\overline{p}_1)} - \left(\ell_1^1 - \lfloor \ell_1^1 \rfloor\right)^2\right\} f^2(\widehat{\mathbb{X}}_1^1)\right]$$
$$= E\left[\left\{\ell_1^1 - \lfloor \ell_1^1 \rfloor - \sum_{j=0}^{R-1} \int_j^{j+1} \int_j^{j+1} 1_{u^1 \in [0, \ell_1^1 - \lfloor \ell_1^1 \rfloor)} 1_{u^2 \in [0, \ell_1^1 - \lfloor \ell_1^1 \rfloor)} du^1 du^2\right\} f^2(\widehat{\mathbb{X}}_1^1)\right]$$
$$= E\left[\left\{\ell_1^1 - \lfloor \ell_1^1 \rfloor - \left(\ell_1^1 - \lfloor \ell_1^1 \rfloor\right)^2\right\} f^2(\widehat{\mathbb{X}}_1^1)\right]$$

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

and by the calculation in (4.9) one finds that

$$E[\xi_1\xi_2] = E\left[\left\{\rho_1^1\rho_1^2 - \left(\ell_1^1 - \lfloor\ell_1^1\rfloor\right)\left(\ell_1^2 - \lfloor\ell_1^2\rfloor\right)\right\} f(\widehat{\mathbb{X}}_1^1) f(\widehat{\mathbb{X}}_1^2)\right]. \tag{4.14}$$

Next, it follows by (4.7,4.4) that

$$\begin{aligned}
&= E\left[\rho_1^1\rho_1^2 f(\widehat{\mathbb{X}}_1^1) f(\widehat{\mathbb{X}}_1^2)\right] \qquad (4.15)\\
&= E\left[\sum_{i\neq k} 1_{\mathbb{U}_k\in[0,\overline{p}_1)} 1_{\mathbb{U}_i\in[\overline{p}_1,\overline{p}_1+\overline{p}_2)} f(\widehat{\mathbb{X}}_1^1) f(\widehat{\mathbb{X}}_1^2)\right]\\
&= E\left[R^2 \frac{(\ell_1^1 - \lfloor\ell_1^1\rfloor)}{R}\frac{(\ell_1^2 - \lfloor\ell_1^2\rfloor)}{R} f(\widehat{\mathbb{X}}_1^1) f(\widehat{\mathbb{X}}_1^2)\right]\\
&- E\left[\sum_{j=0}^{R-1}\int_j^{j+1}\int_j^{j+1} 1_{\left[0,\ell_1^1-\lfloor\ell_1^1\rfloor\right)}(u^1) 1_{\left[\ell_1^1-\lfloor\ell_1^1\rfloor,\ell_1^1-\lfloor\ell_1^1\rfloor+\ell_1^2-\lfloor\ell_1^2\rfloor\right)}(u^2) du^1 du^2 f(\widehat{\mathbb{X}}_1^1) f(\widehat{\mathbb{X}}_1^2)\right]\\
&= E\left[\left(\ell_1^1 - \lfloor\ell_1^1\rfloor\right)\left(\ell_1^2 - \lfloor\ell_1^2\rfloor\right) f(\widehat{\mathbb{X}}_1^1) f(\widehat{\mathbb{X}}_1^2)\right]\\
&- E\left[\left\{1 \wedge \left(\ell_1^1 + \ell_1^2 - \lfloor\ell_1^1\rfloor - \lfloor\ell_1^2\rfloor\right)\left(\ell_1^1 - \lfloor\ell_1^1\rfloor\right) - \left(\ell_1^1 - \lfloor\ell_1^1\rfloor\right)^2\right\} f(\widehat{\mathbb{X}}_1^1) f(\widehat{\mathbb{X}}_1^2)\right].
\end{aligned}$$

Hence, by (4.14) and (4.15)

$$E[\xi_1\xi_2] = -E\left[\left\{1 \wedge \left(\ell_1^1 + \ell_1^2 - \lfloor\ell_1^1\rfloor - \lfloor\ell_1^2\rfloor\right)\left(\ell_1^1 - \lfloor\ell_1^1\rfloor\right) - \left(\ell_1^1 - \lfloor\ell_1^1\rfloor\right)^2\right\} f(\widehat{\mathbb{X}}_1^1) f(\widehat{\mathbb{X}}_1^2)\right] \tag{4.16}$$

and so by (4.16,4.13,4.2) the combined resampling expected conditional variance is:

$$\begin{aligned}
\gamma_1^{CR} &= E\left[\left\{\ell_1^1 - \lfloor\ell_1^1\rfloor - \left(\ell_1^1 - \lfloor\ell_1^1\rfloor\right)^2\right\}\frac{f^2(\widehat{\mathbb{X}}_1^1)}{N}\right]\\
&- \frac{N-1}{N} E\left[\left\{1 \wedge \left(\ell_1^1 + \ell_1^2 - \lfloor\ell_1^1\rfloor - \lfloor\ell_1^2\rfloor\right)\left(\ell_1^1 - \lfloor\ell_1^1\rfloor\right) - \left(\ell_1^1 - \lfloor\ell_1^1\rfloor\right)^2\right\} f(\widehat{\mathbb{X}}_1^1) f(\widehat{\mathbb{X}}_1^2)\right].
\end{aligned}$$

4.1.4. *Stratified Resampling Filter.* Following the Combined Resampling arguments, we find that

$$\begin{aligned}
E[\xi_1^2] &= E\left[\left\{\ell_1^1 - \sum_{j=0}^{N-1}\left(\int_j^{j+1} 1_{u\in[0,\ell_1^1)} du\right)^2\right\} f^2(\widehat{\mathbb{X}}_1^1)\right] \qquad (4.17)\\
&= E\left[\left\{\ell_1^1 - \lfloor\ell_1^1\rfloor - \left(\ell_1^1 - \lfloor\ell_1^1\rfloor\right)^2\right\} f^2(\widehat{\mathbb{X}}_1^1)\right]
\end{aligned}$$

and

$$\begin{aligned}
E[\xi_1\xi_2] &= -E\left[\sum_{j=0}^{R-1}\int_j^{j+1} 1_{[0,\ell_1^1)}(u) du \int_j^{j+1} 1_{\left[\ell_1^1,\ell_1^1+\ell_1^2\right)}(v) dv f(\widehat{\mathbb{X}}_1^1) f(\widehat{\mathbb{X}}_1^2)\right] \qquad (4.18)\\
&= -E\left[\left(\ell_1^1 - \lfloor\ell_1^1\rfloor\right)\left\{\left(\lfloor\ell_1^1\rfloor + 1\right) \wedge \left(\ell_1^1 + \ell_1^2\right) - \ell_1^1\right\} f(\widehat{\mathbb{X}}_1^1) f(\widehat{\mathbb{X}}_1^2)\right]
\end{aligned}$$

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

so

$$\begin{aligned}\gamma_1^{SR} &= E\left[\left\{\ell_1^1 - \lfloor \ell_1^1 \rfloor - \left(\ell_1^1 - \lfloor \ell_1^1 \rfloor\right)^2\right\} \frac{f^2(\widehat{\mathbb{X}}_1^1)}{N}\right] \\ &- \frac{N-1}{N} E\left[\left(\ell_1^1 - \lfloor \ell_1^1 \rfloor\right)\left\{\left(\lfloor \ell_1^1 \rfloor + 1\right) \wedge \left(\ell_1^1 + \ell_1^2\right) - \ell_1^1\right\} f(\widehat{\mathbb{X}}_1^1) f(\widehat{\mathbb{X}}_1^2)\right].\end{aligned}$$

4.2. **Branching Filters.** The change in the unnormalized filter due to the first branch is

$$\frac{1}{N}\sum_{k=1}^{N}\left(\left(\left\lfloor \frac{\widehat{\mathbb{L}}_1^k}{\mathbb{A}_1}\right\rfloor + \rho_1^k\right)\mathbb{A}_1 - \widehat{\mathbb{L}}_1^k\right) f(\widehat{\mathbb{X}}_1^k) 1_{\widehat{\mathbb{L}}_1^k \notin (a_0\mathbb{A}_1, b_0\mathbb{A}_1)}$$

and this translates into the change

$$\frac{1}{N}\sum_{k=1}^{N}\left(\left\lfloor \frac{\widehat{\mathbb{L}}_1^k}{\mathbb{A}_1}\right\rfloor + \rho_1^k - \frac{\widehat{\mathbb{L}}_1^k}{\mathbb{A}_1}\right) f(\widehat{\mathbb{X}}_1^k) 1_{\widehat{\mathbb{L}}_1^k \notin (a_0\mathbb{A}_1, b_0\mathbb{A}_1)}$$

for the normalized filter, which can be compared to the error for the resampled particle filters. Moreover, we can use (4.1,4.2) with

$$\xi_k = \left(\lfloor \ell_1^k \rfloor + \rho_1^k - \ell_1^k\right) f(\widehat{\mathbb{X}}_1^k) 1_{\widehat{\mathbb{L}}_1^k \notin (a_0\mathbb{A}_1, b_0\mathbb{A}_1)}.$$

Furthermore, in both our branching models ρ_1^k is $(\ell_1^k - \lfloor \ell_1^k \rfloor)$-Bernoulli,

$$\begin{aligned}E[\xi_1^2] &= E\left[\left\{\ell_1^1 - \lfloor \ell_1^1 \rfloor - \left(\ell_1^1 - \lfloor \ell_1^1 \rfloor\right)^2, \right\} f^2(\widehat{\mathbb{X}}_1^1) 1_{\ell_1^1 \notin (a_0, b_0)}\right] \\ E\left[\xi_1\xi_2\right] &= E\left[\left(\lfloor \ell_1^1 \rfloor + 1_{\mathbb{U}_1^1 \le \ell_1^1 - \lfloor \ell_1^1 \rfloor} - \ell_1^1\right) f(\widehat{\mathbb{X}}_1^1) 1_{\ell_1^1 \notin (a_0, b_0)}\right. \\ &\times \left.\left(\lfloor \ell_1^2 \rfloor + 1_{\mathbb{U}_1^2 \le \left(\ell_1^2 - \lfloor \ell_1^2 \rfloor\right)} - \ell_1^2\right) f(\widehat{\mathbb{X}}_1^2) 1_{\ell_1^2 \notin (a_0, b_0)}\right].\end{aligned}$$

4.2.1. *Residual Branching.* In this case, the $\mathbb{U}$'s are i.i.d. and independent of everything so the two particle correlation and the expected conditional variance are:

$$\begin{aligned}E\left[\xi_1\xi_2\right] &= 0 \\ \gamma_1^{RB} &= E\left[\left\{\ell_1^1 - \lfloor \ell_1^1 \rfloor - \left(\ell_1^1 - \lfloor \ell_1^1 \rfloor\right)^2\right\} \frac{f^2(\widehat{\mathbb{X}}_1^1)}{N} 1_{\ell_1^1 \notin (a_0, b_0)}\right].\end{aligned}$$

4.2.2. *Combined Branching.* We use stratified random variables in combined branching. (4.3) implies the combined branching correlations:

$$\begin{aligned}E\left[\xi_1\xi_2\right] &= E\left[\frac{R^2}{R(R-1)}\left(\lfloor \ell_1^1 \rfloor + 1_{U \le \left(\ell_1^1 - \lfloor \ell_1^1 \rfloor\right)} - \ell_1^1\right) f(\widehat{\mathbb{X}}_1^1) 1_{\ell_1^1 \notin (a_0, b_0)}\right. \\ &\times \left.\left(\lfloor \ell_1^2 \rfloor + 1_{V \le \left(\ell_1^2 - \lfloor \ell_1^2 \rfloor\right)} - \ell_1^2\right) f(\widehat{\mathbb{X}}_1^2) 1_{\ell_1^2 \notin (a_0, b_0)}\right] \\ &- E\left[\frac{1}{R(R-1)}\sum_{j=0}^{R-1}\left(\lfloor \ell_1^1 \rfloor + 1_{U_j \le \left(\ell_1^1 - \lfloor \ell_1^1 \rfloor\right)} - \ell_1^1\right) f(\widehat{\mathbb{X}}_1^1) 1_{\ell_1^1 \notin (a_0, b_0)}\right. \\ &\times \left.\left(\lfloor \ell_1^2 \rfloor + 1_{V_j \le \left(\ell_1^2 - \lfloor \ell_1^2 \rfloor\right)} - \ell_1^2\right) f(\widehat{\mathbb{X}}_1^2) 1_{\ell_1^2 \notin (a_0, b_0)}\right].\end{aligned} \tag{4.19}$$

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

Here, U, V are independent $[0,1]$-uniform random variables and each U_j, V_j are independent pairs of $\left[\frac{j}{R}, \frac{j+1}{R}\right]$-uniform random variables. However, the first term of (4.19) is zero by independence so

$$\begin{aligned}
&E\left[\xi_1\xi_2\right] \qquad (4.20)\\
&= E\left[\frac{f(\widehat{\mathbb{X}}_1^1)f(\widehat{\mathbb{X}}_1^2)}{R-1}\left\{\frac{\ell_1^1-\lfloor\ell_1^1\rfloor}{R}\sum_{j=0}^{R-1}1_{V_j\le\ell_1^2-\lfloor\ell_1^2\rfloor}+\frac{\ell_1^2-\lfloor\ell_1^2\rfloor}{R}\sum_{j=0}^{R-1}1_{U_j\le\ell_1^1-\lfloor\ell_1^1\rfloor}\right.\right.\\
&-\left.\left.\left(\ell_1^1-\lfloor\ell_1^1\rfloor\right)\left(\ell_1^2-\lfloor\ell_1^2\rfloor\right)-\frac{1}{R}\sum_{j=0}^{R-1}1_{U_j\le\ell_1^1-\lfloor\ell_1^1\rfloor,V_j\le\ell_1^2-\lfloor\ell_1^2\rfloor}\right\}1_{\ell_1^1,\ell_1^2\notin(a_0,b_0)}\right]\\
&= E\left[\frac{f(\widehat{\mathbb{X}}_1^1)f(\widehat{\mathbb{X}}_1^2)}{R-1}\left\{\left(\ell_1^1-\lfloor\ell_1^1\rfloor\right)\left(\ell_1^2-\lfloor\ell_1^2\rfloor\right)-\left(\ell_1^1-\lfloor\ell_1^1\rfloor\right)\wedge\left(\ell_1^2-\lfloor\ell_1^2\rfloor\right)\right\}1_{\ell_1^1,\ell_1^2\notin(a_0,b_0)}\right]
\end{aligned}$$

Hence, the combined branching expected conditional variance is:

$$\begin{aligned}
&\gamma_1^{CB} = E\left[\left[\ell_1^1-\lfloor\ell_1^1\rfloor-\left(\ell_1^1-\lfloor\ell_1^1\rfloor\right)^2\right]\frac{f^2(\widehat{\mathbb{X}}_1^1)}{N}1_{\ell_1^1\notin(a_0,b_0)}\right]\\
&-\frac{N-1}{N}E\left[\frac{f(\widehat{\mathbb{X}}_1^1)f(\widehat{\mathbb{X}}_1^2)}{R-1}1_{\ell_1^1,\ell_1^2\notin(a_0,b_0)}\left\{\frac{\lfloor R\left(\ell_1^1-\lfloor\ell_1^1\rfloor\right)\wedge\left(\ell_1^2-\lfloor\ell_1^2\rfloor\right)\rfloor}{R}-\left(\ell_1^1-\lfloor\ell_1^1\rfloor\right)\left(\ell_1^2-\lfloor\ell_1^2\rfloor\right)\right.\right.\\
&\left.\left.+\frac{1\wedge\{R(\ell_1^1-\lfloor\ell_1^1\rfloor)-\lfloor R(\ell_1^1-\lfloor\ell_1^1\rfloor)\wedge(\ell_1^2-\lfloor\ell_1^2\rfloor)\rfloor\}\{R(\ell_1^2-\lfloor\ell_1^2\rfloor)-\lfloor R(\ell_1^1-\lfloor\ell_1^1\rfloor)\wedge(\ell_1^2-\lfloor\ell_1^2\rfloor)\rfloor\}\wedge1}{R}\right\}\right]
\end{aligned}$$

4.3. **Discussion of Particle Methods.** In all cases, the second term of the expected conditional variance could be positive or negative, depending upon f and how close the particles are. Hence, we look at the first term:

Method	First term of γ_1
Bootstrap	$E\left[\left\{\ell_1^1-\frac{\left(\ell_1^1\right)^2}{N}\right\}\frac{f^2(\widehat{\mathbb{X}}_1^1)}{N}\right]$
Residual Resampling	$E\left[\left\{\ell_1^1-\lfloor\ell_1^1\rfloor-\frac{\left(\ell_1^1-\lfloor\ell_1^1\rfloor\right)^2}{R}\right\}\frac{f^2(\widehat{\mathbb{X}}_1^1)}{N}\right]$
Stratified Resampling	$E\left[\left\{\ell_1^1-\lfloor\ell_1^1\rfloor-\left(\ell_1^1-\lfloor\ell_1^1\rfloor\right)^2\right\}\frac{f^2(\widehat{\mathbb{X}}_1^1)}{N}\right]$
Combined Resampling	$E\left[\left\{\ell_1^1-\lfloor\ell_1^1\rfloor-\left(\ell_1^1-\lfloor\ell_1^1\rfloor\right)^2\right\}\frac{f^2(\widehat{\mathbb{X}}_1^1)}{N}\right]$
Residual Resampling	$E\left[\left\{\ell_1^1-\lfloor\ell_1^1\rfloor-\left(\ell_1^1-\lfloor\ell_1^1\rfloor\right)^2\right\}\frac{f^2(\widehat{\mathbb{X}}_1^1)}{N}1_{\ell_1^1\notin(a_0,b_0)}\right]$
Combined Resampling	$E\left[\left\{\ell_1^1-\lfloor\ell_1^1\rfloor-\left(\ell_1^1-\lfloor\ell_1^1\rfloor\right)^2\right\}\frac{f^2(\widehat{\mathbb{X}}_1^1)}{N}1_{\ell_1^1\notin(a_0,b_0)}\right]$

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

We listed these in order of improvement: $\ell_1^1 - \lfloor \ell_1^1 \rfloor - (\ell_1^1 - \lfloor \ell_1^1 \rfloor)^2$ can not be more than $\frac{1}{4}$ and is expected to be small. Notice the Stratified Resampling method achieves this even though the residual technique of deterministically keeping as many particles is not used. Stratified and Combined Resampling are clearly superior to Residual Resampling and Bootstrap. However, we do not recommend either but rather strongly suggest switching to branching algorithms like those introduced in this paper. The two branching methods have the factor $1_{\ell_1^1 \notin (a_0, b_0)}$, which when a_0, b_0 are chosen intelligently can get rid of most of the resampling/branching noise. Naturally, the speed improvements in switching to branching is also important. The negative correlations introduced in the Combined Branching will help the number of particles stay more constant and thereby improve performance over the Residual Branching.

5. Comparison of Tracking and Model Selection

Effective parallelization of resampled particle filters is difficult (see e.g. Vergé et. al. (2013) for a good attempt). This is reason enough to choose our branching particle filters over the resampled ones as they are fundamentally more parallelizable (as will be shown in future work). However, we can also consider all algorithms with single-processor implementations on tracking problems and on model selection. The rest of this section is organized as follows: We first introduce the two simple problems, our *Test* and *Range-Only* problems, that will be used for comparison purposes. Then, we compare the various resampled particle systems discussed above on these problems. Next, we compare the worst of our branching algorithms to the best resampled particle system discussed above and show even this most basic branching algorithm significantly outperforms all resampled particle systems. Finally, we compare all our branching algorithms to determine which variation performs the best. For consistency, all results herein are either a *typical path* or an *average over 200 different sample paths* of *35 time steps*.

5.1. **Test Model.**

The *Test Model* refers to the scalar signal and observation pair:

$$X_n = 0.95X_{n-1} + 0.3W_n, \quad Y_n = X_{n-1} + V_n,$$

where X_0, $\{W_n\}$ and $\{V_n\}$ are independent with standard Cauchy distribution. This is a linear, non-Gaussian filtering problem. The Kalman filter does not apply since the noise is Cauchy. Indeed, conditional expectations of state do not exist since the noise is heavy-tailed. However, this problem is in other respects simple.

For model selection, we introduce alternative models and show that we select the correct one. We keep most of the *Test Model* the same and just vary two coefficients:

Model Number	Signal Equation	Observation Equation
-2	$X_n = 0.93X_{n-1} + 0.28W_n$	$Y_n = X_{n-1} + V_n$
-1	$X_n = 0.94X_{n-1} + 0.29W_n$	$Y_n = X_{n-1} + V_n$
0	$X_n = 0.95X_{n-1} + 0.3W_n$	$Y_n = X_{n-1} + V_n$
1	$X_n = 0.96X_{n-1} + 0.31W_n$	$Y_n = X_{n-1} + V_n$
2	$X_n = 0.97X_{n-1} + 0.32W_n$	$Y_n = X_{n-1} + V_n$

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

Hence, the real model is model 0, the null model.

5.2. **Range Only Model.** The *Range Only Model* refers to the four dimensional signal and scalar observation model:

Position	*Velocity*
$X_n = \alpha X_{n-1} + U_{n-1} + 0.3\alpha_n$	$U_n = 0.95U_{n-1} + \gamma_{n-1}$
$Z_n = \alpha Z_{n-1} + V_{n-1} + 0.3\beta_n$	$V_n = 0.95V_{n-1} + \theta_{n-1}$

$$\alpha = 0.5, \quad Y_n = \sqrt{X_{n-1}^2 + Z_{n-1}^2} + 0.1\psi_n,$$

TABLE 1. Range Only Model

where X_0, Z_0, U_0, V_0, $\{\gamma_n\}$, $\{\theta_n\}$, $\{\alpha_n\}$, $\{\beta_n\}$, $\{\psi_n\}$ are independent. X_0, Z_0 have 10 times the standard Cauchy distribution and U_0, V_0 have 5 times the standard Normal distribution. Signal noise sources γ_n and θ_n have standard normal distribution while α_n and β_n have standard Cauchy distribution. ψ_n is the observation noise with standard Cauchy distribution. This is a nonlinear problem but otherwise simple.

The idea of this model comes from radar detection. Suppose that there is a radar station at the origin in the plane, (X_n, Z_n) describes the position of a ship and (U_n, V_n) its velocity. The radar produces a noise-corrupted distance observation between the ship and itself, which is Y_n in our model. The objective of this problem is to estimate the state of the ship using the (back) observations.

For model-selection alternative models, we will keep most of the *Range Only Model* the same and just vary the coefficient in front of X_{n-1} and Z_{n-1} slightly:

Model Number	Signal Parameter	Observation Equation
-2	$\alpha = 0.48$	$Y_n = \sqrt{X_{n-1}^2 + Z_{n-1}^2} + 0.1\psi_n$
-1	$\alpha = 0.49$	$Y_n = \sqrt{X_{n-1}^2 + Z_{n-1}^2} + 0.1\psi_n$
0	$\alpha = 0.50$	$Y_n = \sqrt{X_{n-1}^2 + Z_{n-1}^2} + 0.1\psi_n$
1	$\alpha = 0.51$	$Y_n = \sqrt{X_{n-1}^2 + Z_{n-1}^2} + 0.1\psi_n$
2	$\alpha = 0.52$	$Y_n = \sqrt{X_{n-1}^2 + Z_{n-1}^2} + 0.1\psi_n$

Hence, the real model is model 0 with the others differing slightly through α.

5.3. **Comparison within Resampled Particle Systems.** First, we compare resampled particle systems based on both error (of root-mean-square type between positional tracking estimation and the real value) and execution time.

For our *Test Model*, the error is defined as

$$error = \sqrt{\frac{1}{n}\sum_{k=1}^{n}(\pi_k^N(f) - f(X_k))^2}, \quad f(x) = \begin{cases} 30 & : \quad x > 30 \\ x & : \quad -30 \le x \le 30 \\ -30 & : \quad x < -30 \end{cases}.$$

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

where π_k^N is the filter approximation at time instant k and N particles and X_k is the signal. For our *Range Only Model*, the error is

$$error = \frac{1}{n}\sum_{k=1}^{n}\sqrt{(\pi_k^N(g_x) - g(X_k))^2 + (\pi_k^N(g_z) - g(Z_k))^2},$$

where π_k^N is the normalized filter approximation at time instant k and N initial particles, X_k, Z_k are the positional components of the real signal,

$$g(x) = \begin{cases} 1000 & : \quad x > 1000 \\ x & : \quad -1000 \leq x \leq 1000 \\ -1000 & : \quad x < -1000 \end{cases} .$$

and g_x, g_z denote g applied to the x and z (positional) components of the signal.

The result for error of *Test Model* and *Range Only Model* are shown in Table 2 and Table 3 respectively for the algorithms defined in Section 2.

Particle Number N	**100**	**400**	**2000**	**10000**
Bootstrap	8.5360	7.8759	6.9974	5.2674
Residual Resampling	7.4110	6.6604	5.8705	5.2170
Stratified Resampling	7.3761	6.5142	5.8674	5.2597
Systematic Resampling	7.3743	6.5818	5.8898	5.2655
Combined Resampling	7.3752	6.5018	5.8723	5.2192

TABLE 2. Average Error of Test Model

All four improved resampling methods show a significant improvement over the bootstrap in the *Test Model* and the *Range Only Model*. All these methods approach the optimal filter as the number of particles increases. In difficult real life problems, one often has to limit the number of particles for computational reasons and performance with lower particle numbers are most important.

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

Particle Number N	**500**	**2000**	**10000**	**50000**
Bootstrap	54.0456	48.9549	47.8813	46.7844
Residual Resampling	52.3962	48.5324	47.6532	46.0332
Stratified Resampling	49.7957	47.9813	46.2924	45.6042
Systematic Resampling	51.3571	48.6998	47.3571	46.0677
Combined Resampling	51.7296	48.0274	47.2888	45.9658

TABLE 3. Average Error of Range Only Model

We should not just pick the method with lowest error as speed is also important. Average execution time results are shown in Table 4 and Table 5 respectively.

Particle Number N	**100**	**400**	**2000**	**10000**
Bootstrap	0.0084	0.0363	0.1680	0.9321
Residual Resampling	0.0081	0.0331	0.1557	0.8189
Stratified Resampling	0.0078	0.0295	0.1496	0.7947
Systematic Resampling	0.0065	0.0261	0.1123	0.5398
Combined Resampling	0.0080	0.0335	0.1302	0.6976

TABLE 4. Average Execution Times for Test Model

Residual and combined resampling preserve some particles without resampling saving some of the resampling computations. The stratified, combined and systematic resampling, save computations related to ordering the uniform random variables in the bootstrap method. For simple signal models (like Test), a large portion of the time is consumed generating and ordering the uniform resampling random numbers. This is efficiently done with stratification so it is reasonable that stratified, combined and systematic method can improve the speed of the *Test Model* greatly.

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

Particle Number N	500	2000	10000	50000
Bootstrap	0.0812	0.2597	1.1837	5.8093
Residual Resampling	0.0581	0.2128	1.0122	5.0141
Stratified Resampling	0.0735	0.2618	1.1844	6.0746
Systematic Resampling	0.0602	0.1932	1.0381	4.5857
Combined Resampling	0.0823	0.2127	1.0617	5.7658

TABLE 5. Average Execution Times for Range Only Model

With slightly larger signals such as our *Range Only Model*, which has a four dimensional signal, a lot of time is spent copying particles. Thus, the execution time may depend more on how many particles need to be copied or resampled. Hence, it is also reasonable that residual and combined methods, which reduce the number of particles resampled, can improve execution time a lot. Naturally, the systematic method is very fast as it only uses one uniform random variable.

To combine performance and speed, we define the "Bootstrap Factor" as:

$$Bootstrap \quad Factor = \frac{t_{bootstrap}}{t},$$

where $t_{bootstrap}$ and t are the execution times for the bootstrap and method of interest to reach a fixed error. We use $error = 5.0$ in the *Test Model* and $error = 46.0$ in the *Range Only Model*, then show the minimum particle number, execution time and "Bootstrap Factor" for both in Tables 6 and 7.

	N	Time	Bootstrap Factor
Bootstrap	20000	1.9152	1
Residual Resampling	20000	1.6399	1.1679
Stratified Resampling	10000	0.7947	2.4099
Systematic Resampling	25000	1.5582	1.2291
Combined Resampling	10000	0.6976	2.7454

TABLE 6. Bootstrap Factor of Test Model with fixed $Error = 5.0$

The Bootstrap Factor compares speed of a method to the bootstrap filter for a given performance, combining accuracy and efficiency factors. Combined Resampling is the best method for both models with Bootstrap Factors of 2.7454 and 6.5116. However, *every* branching algorithm will significantly outperform this.

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

	N	Time	Bootstrap Factor
Bootstrap	60000	6.9134	1
Residual Resampling	50000	5.0141	1.3788
Stratified Resampling	10000	1.1844	5.8522
Systematic Resampling	50000	4.5857	1.5076
Combined Resampling	10000	1.0617	6.5116

TABLE 7. Bootstrap Factor of Range Only Model with fixed $Error = 46.0$

5.4. **Comparison between Branching and Resampled Particle Systems.** In this section, we compare the bootstrap and best resampled particle system to residual branching, the most basic branching system. We show that our residual branching can improve both performance and execution time. For now, we define $(a_n, b_n) = (1/r, r)$, where $r \in [1, \infty]$, and refer to r as the resampling parameter. All particles will resample when $r = 1$, which we call complete resampling. No particle will resample when $r = \infty$, which means we have the weighted particle filter. We found a good fixed choice of r for the *Test Model* was 2.25 and for the *Range Only Model* was 5. (Later, we will explore better methods with state-dependent r.)

The error comparison is shown in Tables 8 and 9. Residual branching is much better than the bootstrap and even the best resampled system, combined resampling.

Particle Number N	**100**	**400**	**2000**	**10000**
Bootstrap	8.5360	7.8759	6.9974	5.2254
Combined Resampling	7.3752	6.5018	5.8723	5.2192
Residual Branching	5.4284	4.9177	4.6479	4.5395

TABLE 8. Average Error of Test Model

Particle Number N	**500**	**2000**	**10000**	**50000**
Bootstrap	54.0456	48.9549	47.8813	46.7844
Combined Resampling	51.7296	48.0274	47.2888	45.9658
Residual Branching	46.8439	45.9416	45.6183	45.0748

TABLE 9. Average Error of Range Only Model

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

Speed is compared in Tables 10 and 11. Branching is fastest for both models.

Particle Number N	**100**	**400**	**2000**	**10000**
Bootstrap	0.0084	0.0363	0.1680	0.9321
Combined Resampling	0.0080	0.0335	0.1302	0.6976
Residual Branching	0.0060	0.0210	0.1153	0.4672

TABLE 10. Average Execution Time of Test Model

Particle Number N	**500**	**2000**	**10000**	**50000**
Bootstrap	0.0812	0.2597	1.1837	5.8093
Combined Resampling	0.0823	0.2127	1.0617	5.7658
Residual Branching	0.0644	0.1883	0.9203	4.4107

TABLE 11. Average Execution Time of Range Only Model

To evaluate the advantage of branching on both performance and speed, we provide the Bootstrap Factor in Tables 12 and 13. In the *Test Model*, residual branching is 91.2 and 33.2 times better than bootstrap and combined resampled respectively. In *Range Only Model*, the improvement is also significant at 36.7148 and 5.64. Our better branching algorithms will be shown below to outperform yet a lot more.

	N	**Time**	**Bootstrap Factor**
Bootstrap	20000	1.9152	1
Combined Resampling	10000	0.6976	2.7454
Residual Branching	400	0.0210	91.2000

TABLE 12. Bootstrap Factor of Test Model with Error 5.0

	N	**Time**	**Bootstrap Factor**
Bootstrap	60000	6.9134	1
Combined Resampling	10000	1.0617	6.5116
Residual Branching	2000	0.1883	36.7148

TABLE 13. Bootstrap Factor of Range Only Model with Error 46

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

Model selection ability is also extremely important. For comparison purposes, we fix the initial number of particles to be $N = 10,000$ for all model selection experiments and show the execution time for model selection in Table 14. Residual branching is the fastest for model selection as it was for tracking. Indeed, branching has another small inherent advantage here since model selection is based upon the unnormalized filter, which is already computed in the branching methods.

Model	Test Model	Range Only Model
Bootstrap	1.213	1.613
Combined Resampling	0.960	1.602
Residual Branching	0.736	1.347

TABLE 14. Average Execution Time of Model Selection

Inasmuch as the performance results are very similar for both the Test and Range Only models, we just demonstrate these three algorithm on our *Range Only Model*. We define $Bayes\ Factor = \frac{\sigma^0(1)}{\sigma^k(1)}$, where $k \in \{-2, -1, 0, 1, 2\}$ is the index for the different models described in Sections 5.1 and 5.2. All three algorithms select the correct model rather convincingly. It appears from these single-outcome pictures that bootstrap had the hardest time distinguishing models, Combined Resampling distinguished the correct model from model 1 the best, while Residual Branching distinguished the other three models best.

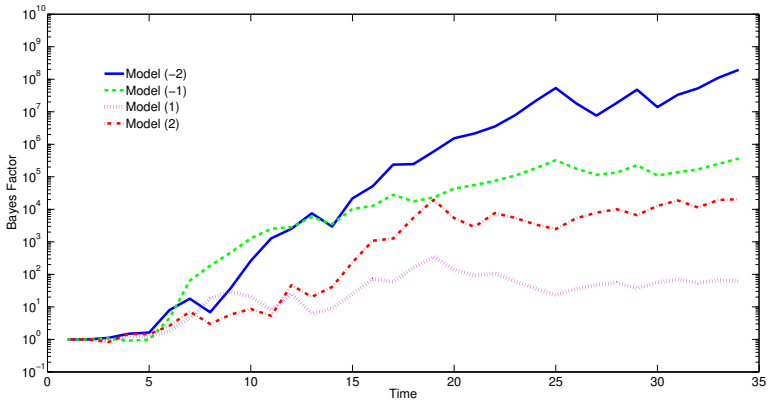

FIGURE 5. Bootstrap Model Selection of *Range Only Model*

Typically, bootstrap has the most difficult time distinguishing models and residual branching is slightly better at distinguishing models than combined resampling.

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

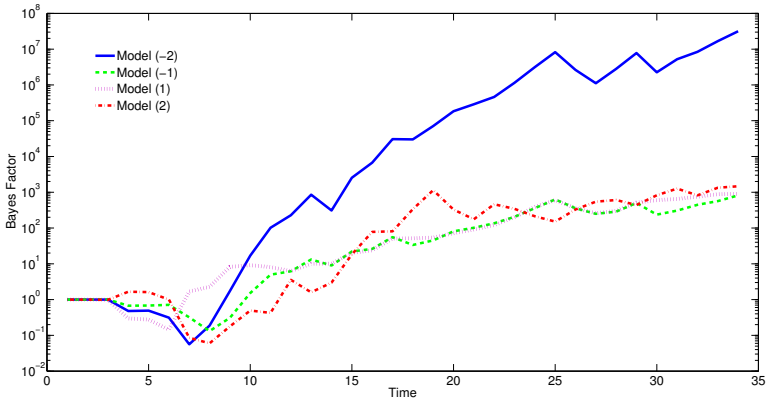

FIGURE 6. Combined Resampling Model Selection of *Range Only Model*

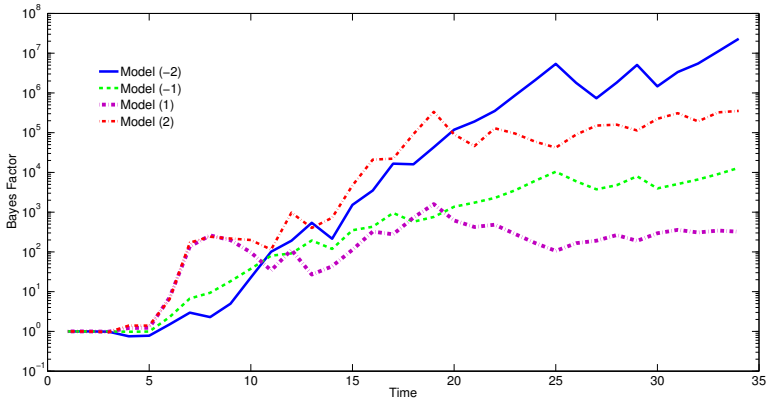

FIGURE 7. Residual Branching Model Selection of *Range Only Model*

5.5. **Comparison within Branching Particle Systems.** There are many ways to produce unbiased branching filters. The residual branching and combined branching algorithms, introduced in Sections 3.1 and 3.2, are two of the simplest to implement. Hitherto, we have taken a_n and b_n in these algorithms constant. However, the performance and speed can improve if a_n and b_n depend on the system state at time step n. Hence, now we consider a dynamic r_n such that $a_n = \frac{1}{r_n}$ and $b_n = r_n$ will depend on system state in two different ways. We call these two implementations Dynamic Branching and Effective Particle Branching as mentioned in Section 3.3.

We continue comparing error, speed and "Bootstrap Factor" but now within branching particle systems. First, we consider simple residual branching and combined branching on our test model and show combined is much better than residual and hence two levels above the resampled particle systems.

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

Particle Number N	**100**	**400**	**2000**	**10000**
Residual Branching	5.4284	4.9177	4.6479	4.5395
Combined Branching	5.1690	4.5548	4.2728	4.1494

TABLE 15. Average Error of Test Model

The error, execution times and bootstrap factors are in Tables 15, 16 and 17.

Particle Number N	**100**	**400**	**2000**	**10000**
Residual Branching	0.0060	0.0210	0.1153	0.4672
Combined Branching	0.0079	0.0312	0.1063	0.5010

TABLE 16. Average Execution Time of Test Model

	N	**Time**	**Bootstrap Factor**
Residual Branching	400	0.0210	91.2100
Combined Branching	150	0.0085	225.3176

TABLE 17. Bootstrap Factor of Test Model with fixed Error 5.0

It is clear that the combined method is superior to the Residual one on the Test model. Insomuch as the conclusions to be formed on the Test and Range Only models would be very similar, we conserve space and only present our experimental results on the dynamic and effective particle methods for the Range Only model.

Next, we apply residual branching (with $r = 5$), combined branching (with $r = 5$), dynamic branching and effective particle branching to our Range Only Model. It turns out that this good choice of $r = 5$ translates into branching about three quarters of the time in the residual and combined methods. Also, since combined branching (where stratified random numbers are used) outperforms residual branching, we use both residual and stratified techniques within dynamic and effective particle branching. For clarity, these approaches differ from combined branching by replacing a fixed r with a state dependent r_n as defined in Section 3.3.

For *dynamic branching*, we set q to be different value and find the best c, which means the minimum error, and show the results in Table 18. It shows that the resampling percentage is always around 76%, which is a similar amount as used in with the good choice of static r in the residual and combined branching filters.

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

q	**c**	**Error**	**resampling percent**
0.5	0.80	46.3298	76.19%
1.0	0.60	45.8457	76.21%
1.5	0.36	45.9732	76.51%
2.0	0.25	46.3665	76.52%

TABLE 18. Average Error of dynamic branching method with 500 particles

However, this table demonstrates that the conditions for branching, not just the overall amount of branching, affect error. The error is minimal with $q = 1$, which corresponds to a constant amount of branching regardless of system entropy. The Residual and Combined methods would correspond to the case of more branching when there is more entropy and the case $q = 2$ would correspond to the case of more branching when there is less entropy. Indeed, we will see below that this dynamic branching with $q = 1$ beats combined branching at all particle numbers considered.

For effective particle branching, we consider the following c^{noneff} and c^{eff} choices:

c^{eff}	c^{non}	**Error**	c^{eff}	c^{non}	**Error**	c^{eff}	c^{non}	**Error**
1	2	64.5101	1	16	45.7550	2	8	45.9141
2	1	46.2599	16	1	46.4342	8	2	47.7404
1	4	45.9427	1	32	48.2642	2	16	45.8674
4	1	53.0337	32	1	50.0621	16	2	53.2428
1	8	45.9082	2	4	56.0850	2	32	46.5672
8	1	46.1202	4	2	49.2555	32	2	54.7030

TABLE 19. Average Error with Different c^{eff} and c^{noneff} and 500 particles

The results in Table 19 show that $c^{eff} = 1$ and $c^{noneff} = 16$ for r_n in (3.2) is the best choice to minimize error. This means that more resampling would be done when there are fewer effective particles for the Range Only problem.

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

We compare residual, combined, dynamic and effective particle branching errors in Table 20. It shows that the effective particle method performs slightly better than the dynamic method and both of these are considerably better than the combined branching. We conclude that state-dependent branching is worthwhile.

Particle Number N	**500**	**2000**	**10000**	**50000**
Residual Branching	46.8439	45.9416	45.6183	45.0748
Combined Branching	45.9928	45.5704	45.1756	44.9879
Dynamic Branching	45.8457	45.4156	44.9498	44.5768
Effective Particle	45.7550	45.4107	44.8365	44.5617

TABLE 20. Average Error of Range Only Model

Before we declare the Effective Particle Branching method to be the best, we need to consider execution time. The results are shown in Table 21.

Particle Number N	**500**	**2000**	**10000**	**50000**
Residual Branching	0.0644	0.1883	0.9203	4.4107
Combined Branching	0.0480	0.2090	0.9310	4.5120
Dynamic Branching	0.0658	0.2155	1.1496	4.9783
Effective Particle Branching	0.0693	0.2451	1.1527	5.0135

TABLE 21. Average Execution Time of Range Only Model

Since dynamic branching is faster than effective particle branching while the performance of effective particle branching is slightly better, we need to consider Bootstrap Factor. Table 22 shows dynamic branching is best on a performance per computation point of view. Still, Combined branching and Effective Particle branching have the advantages of simplicity and best (non-time normalized) performance.

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

	N	Time	Bootstrap Factor
Residual Branching	2000	0.1883	36.7148
Combined Branching	500	0.0480	144.0292
Dynamic Branching	350	0.0412	167.8009
Effective Particle Branching	320	0.0465	148.6753

TABLE 22. Bootstrap Factor of Range Only with Error 46.0

To demonstrate the significant improvement of branching systems over resampled systems, we show typical *Test Model* and *Range Only Model* error versus time.

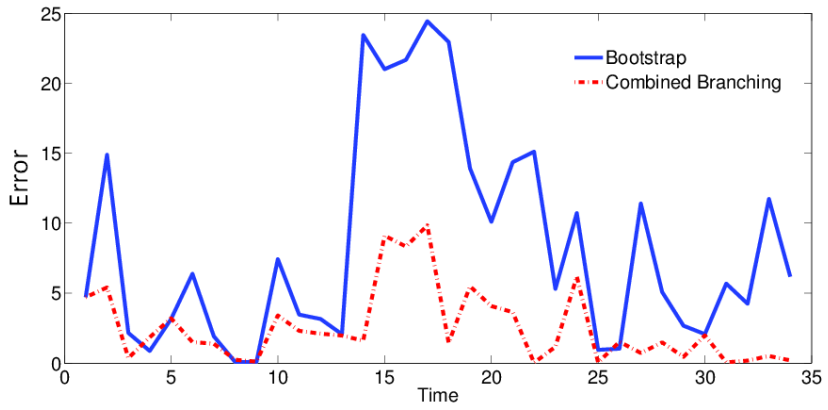

FIGURE 8. typical case in *Test Model*

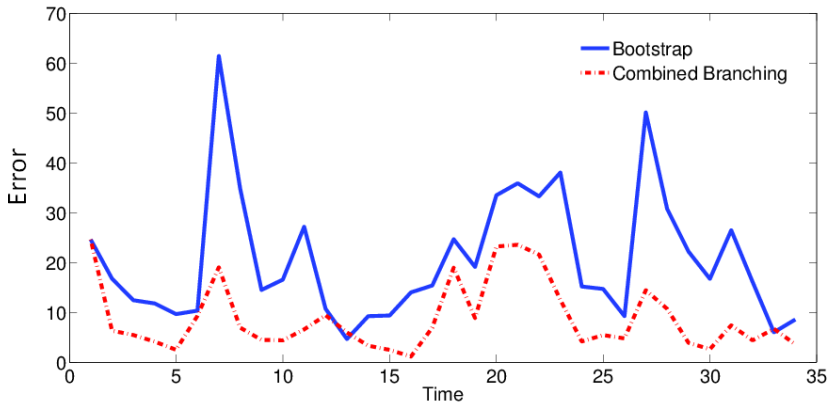

FIGURE 9. typical case in *Range Only Model*

6. CONCLUSIONS

Based upon our experimental and theoretical results, we suggest the following:

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

(1) There are branching particle methods that do not suffer terribly from wild particle swings.
(2) There are branching particle methods whose tracking performance and execution times can compare most favorably to the traditional resampled particle systems that have widespread appeal.
(3) Practioners should now consider the Combined, Dynamic and Effective Particle branching algorithms introduced herein.
(4) The Bootstrap Factor defined herein is a reasonable way to compare particle filtering methods.
(5) Branching particle filters also compare favorably on model selection problems and have the added advantage of using the unnormalized filter for ease of computing Bayes factor.

7. Appendix: Proof of Theorem 2

Proof. **Initial Setup:** Let $p_{l+1}(j)$ denote the parent of the j^{th} particle at time $l+1$, $\mathbb{W}_l^i = \alpha_l(\mathbb{X}_{l-1}^i)$, $\mathbb{H}_{l+1}^k = \{\widehat{\mathbb{L}}_{l+1}^k \notin (a_l\mathbb{A}_{l+1}, b_l\mathbb{A}_{l+1})\}$ and $\chi_l^k = \frac{\mathbb{W}_{l+1}^k\mathbb{L}_l^k}{\mathbb{A}_{l+1}^N}1_{\mathbb{H}_{l+1}^k} + 1_{(\mathbb{H}_{l+1}^k)^C}$. Since

$$0 < \inf_{x\in E}\frac{g(Y_l - h(x))}{g(Y_l)},\ \sup_{x\in E}\frac{g(Y_l - h(x))}{g(Y_l)} < \infty$$

there is a $C = C(Y_1, ..., Y_n) > 1$ such that

$$\frac{1}{C} \leq \mathbb{W}_l^i \leq C\ \forall 1 \leq i \leq \mathbb{N}_{l-1}; 1 \leq l \leq n; N \geq 1. \tag{7.1}$$

For $l \geq 1$, we define $\upsilon_C(l), \tau_C(l), \mathbb{D}_l^N$ recursively by

$$\upsilon_C(l) = C\upsilon_C(l-1)\tau_C(l-1), \quad \text{subject to } \upsilon_C(0) = 1, \tag{7.2}$$
$$\tau_C(l) = 2\tau_C(l-1)(C\upsilon_C(l)\upsilon_C(l-1)+1) \quad \text{subject to } \tau_C(0) = 1, \tag{7.3}$$
$$\mathbb{D}_l^N = \left\{\frac{1}{\tau_C(l)} \leq \frac{\mathbb{N}_l}{N} \leq \tau_C(l)\right\} \cap \mathbb{D}_{l-1}^N \text{ subject to } \mathbb{D}_0^N = \Omega. \tag{7.4}$$

Clearly, $\mathbb{D}_l^N \in \mathcal{F}_l^{\mathbb{X}\mathbb{U}\mathbb{V}}$. Now, recall from the Residual branching algorithm that

$$\mathbb{A}_{l+1}^N = \frac{1}{N}\sum_{i=1}^{\mathbb{N}_l}\mathbb{W}_{l+1}^i\mathbb{L}_l^i \tag{7.5}$$
$$\mathbb{A}_{l+1}^N \wedge \mathbb{W}_{l+1}^{p_{l+1}(j)}\mathbb{L}_l^{p_{l+1}(j)} \leq \mathbb{L}_{l+1}^j \leq \mathbb{A}_{l+1}^N \vee \mathbb{W}_{l+1}^{p_{l+1}(j)}\mathbb{L}_l^{p_{l+1}(j)} \tag{7.6}$$
$$\mathbb{N}_{l+1}^k \leq \left[\frac{\mathbb{L}_l^k\mathbb{W}_{l+1}^k}{\mathbb{A}_{l+1}^N} + 1\right] \tag{7.7}$$

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

for $j = 1, ..., \mathbb{N}_{l+1}$. These imply by induction and (7.1) that

$$\begin{aligned}
\frac{1}{\upsilon_C(l+1)} &\leq \mathbb{A}_{l+1}^N \leq \upsilon_C(l+1) && (7.8)\\
\frac{1}{\upsilon_C(l+1)} &\leq \mathbb{L}_{l+1}^i \leq \upsilon_C(l+1) \quad \forall i \in \{1, ..., \mathbb{N}_{l+1}\} && (7.9)\\
\frac{1}{C\upsilon_C(l)} &\leq \mathbb{W}_{l+1}^i \mathbb{L}_l^i \leq C\upsilon_C(l) \quad i \in \{1, ..., \mathbb{N}_{l+1}\} && (7.10)\\
\mathbb{N}_{l+1}^k &\leq M_C(l+1) \doteq \upsilon_C(l+1)\upsilon_C(l)C + 1 \quad \forall k \in \{1, 2, ..., \mathbb{N}_l\} && (7.11)
\end{aligned}$$

on $\mathbb{D}_l^N$ for all $l = 0, 1, 2..., n$.

Base Case: $\{\mathbb{N}_1^k\}$ are bounded by $M_C(1)$ (since $\mathbb{D}_0^N = \Omega$) and conditionally independent (since the $\mathbb{U}_1^k$'s are independent) so Hoeffding's inequality applies to find

$$\begin{aligned}
& Q^Y\left(\left|\frac{1}{N}\sum_{k=1}^N \left[\mathbb{N}_1^k - \left[\frac{\mathbb{W}_1^k}{\mathbb{A}_1^N} 1_{\mathbb{H}_1^k} + 1_{(\mathbb{H}_1^k)^C}\right]\right]\right| > t \middle| \mathcal{F}_0^X \vee \mathcal{F}_1^V\right) && (7.12)\\
\leq\ & 2\exp\left(-\frac{2Nt^2}{M_C^2(1)}\right) \text{ a.s.}
\end{aligned}$$

Next, by (7.1), (7.8), (7.2), (7.3) and (7.12)

$$\begin{aligned}
& Q^Y\left(\left\{\frac{1}{\tau_C(1)} \leq \frac{1}{N}\sum_{k=1}^N \mathbb{N}_1^k \leq \tau_C(1)\right\}\right) && (7.13)\\
\geq\ & Q^Y\left(\left\{\frac{2}{\tau_C(1)} \leq \frac{1}{N}\sum_{k=1}^N \left[\frac{\mathbb{W}_1^k}{\mathbb{A}_1^N} 1_{\mathbb{H}_1^k} + 1_{(\mathbb{H}_1^k)^C}\right] \leq \frac{\tau_C(1)}{2}\right\}\right)\\
-\ & E^Y\left[Q^Y\left(\frac{1}{N}\left|\sum_{k=1}^N \left[\mathbb{N}_1^k - \left[\frac{\mathbb{W}_1^k}{\mathbb{A}_1^N} 1_{\mathbb{H}_1^k} + 1_{(\mathbb{H}_1^k)^C}\right]\right]\right| > \frac{1}{\tau_C(1)} \middle| \mathcal{F}_0^{\mathbb{X}} \vee \mathcal{F}_1^{\mathbb{V}}\right)\right]\\
\geq\ & 1 - 2\exp\left(-\frac{2N}{M_C^2(1)\tau_C^2(1)}\right).
\end{aligned}$$

Inductive Step: Suppose that

$$Q^Y(\mathbb{D}_l^N) \geq 1 - 2l\exp\left(-\frac{2N}{M_C^2(l)\tau_C^2(l)\tau_C^2(l-1)}\right), \qquad (7.14)$$

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

which is true when $l = 1$. Then, it follows by (7.10), (7.8) (7.3) and (7.2) that

$$\begin{aligned} & Q^Y\left(\left\{\frac{1}{\tau_C(l+1)} \le \frac{1}{N}\sum_{k=1}^{\mathbb{N}_l} \mathbb{N}_{l+1}^k \le \tau_C(l+1)\right\} \cap \mathbb{D}_l^N\right) \qquad (7.15) \\ \ge\ & Q^Y\left(\left\{\frac{2}{\tau_C(l+1)} \le \frac{1}{N}\sum_{k=1}^{\mathbb{N}_l} \chi_l^k \le \frac{\tau_C(l+1)}{2}\right\} \cap \mathbb{D}_l^N\right) \\ -\ & Q^Y\left(\left\{\left|\frac{1}{N}\sum_{k=1}^{\mathbb{N}_l}\left[\mathbb{N}_{l+1}^k - \chi_l^k\right]\right| > \frac{1}{\tau_C(l+1)}\right\} \cap \mathbb{D}_l^N\right) \\ \ge\ & Q^Y(\mathbb{D}_l^N) - Q^Y\left(\left\{\left|\frac{1}{N}\sum_{k=1}^{\mathbb{N}_l}\left[\mathbb{N}_{l+1}^k - \chi_l^k\right]\right| > \frac{1}{\tau_C(l+1)}\right\} \cap \mathbb{D}_l^N\right). \end{aligned}$$

However, we have by the independence of the $\mathbb{U}$'s in the Residual branching algorithm, (7.11) and Hoeffding's inequality that

$$\begin{aligned} & Q^Y\left(\left|\frac{1}{N}\sum_{k=1}^{\mathbb{N}_l}\left[\mathbb{N}_{l+1}^k - \chi_l^k\right]\right| > t \middle| \mathcal{F}_\infty^{\mathbb{X},\mathbb{V}} \vee \mathcal{F}_l^{\mathbb{U}}\right) \qquad (7.16) \\ =\ & Q^Y\left(\left|\frac{1}{\mathbb{N}_l}\sum_{k=1}^{\mathbb{N}_l}\left[\mathbb{N}_{l+1}^k - \chi_l^k\right]\right| > \frac{N}{\mathbb{N}_l}t \middle| \mathcal{F}_\infty^{\mathbb{X},\mathbb{V}} \vee \mathcal{F}_l^{\mathbb{U}}\right) \\ \le\ & 2\exp\left(-\frac{2Nt^2}{M_C^2(l+1)\tau_C^2(l)}\right) \quad \text{on } \mathbb{D}_l^N \end{aligned}$$

so by (7.15), (7.14) and (7.16) with $t = \frac{1}{\tau_C(l+1)}$

$$\begin{aligned} & Q^Y\left(\left\{\tau_C(l+1) \le \frac{1}{N}\sum_{k=1}^{\mathbb{N}_l} \mathbb{N}_{l+1}^k \le \tau_C(l+1)\right\} \cap \mathbb{D}_l^N\right) \qquad (7.17) \\ \ge\ & 1 - 2(l+1)\exp\left(-\frac{2N}{M_C^2(l+1)\tau_C^2(l+1)\tau_C^2(l)}\right). \end{aligned}$$

Conclusion: The result follows by induction, (7.4), (7.8) and (7.9). □

References

Ballantyne, D. J., Chan, H. Y. and Kouritzin, M. A. (2000). A novel branching particle method for tracking. In *Signal and Data Processing of Small Targets* (O. E. Drummond, ed.) SPIE 277-287.

Cappe, O., Godsill, S. J., and Moulines, E. (2007). An overview of existing methods and recent advances in sequential Monte Carlo. *Proceedings of the IEEE* **95**, 899-924.

Carpenter, J., Clifford, P. and Fearnhead, P. (1999). An Improved Particle Filter for Nonlinear Problems. *IEE Proc. Radar Sonar Navigation* **146**, 2-7.

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015

Crisan, D. and Lyons, T. (1997). Nonlinear filtering and measure valued processes. *Prob. Theory Related Fields* **109**, 217-244.

Del Moral P., Doucet A., and Jasra A. (2012). On adaptive resampling strategies for sequential Monte Carlo methods. *Bernoulli* **18**, 252-278.

Del Moral, P., Kouritzin, M.A., and Miclo, L. (2001). On a class of discrete generation interacting particle systems. *Electronic Journal of Probability* **6**: Paper No. 16, 26 p.

Del Moral P. and Miclo L. (2000). Branching and Interacting Particle Systems Approximations of Feynman-Kac Formulae with Applications to Non-Linear Filtering. Séminaire de Probabilités XXXIV, Ed. J. Azéma, M. Emery, M. Ledoux and M. Yor. *Lecture Notes in Mathematics*, Springer-Verlag Berlin, Vol. 1729, 1-145.

Douc, R., Cappé, O. and Moulines, E. (2005). Comparison of resampling schemes for particle filtering. In *Proceedings of the* 4^{th} *International Symposium on Image and Signal Processing and Analysis* 64-69.

Doucet, A., Godsill, S.J. and Andrieu, C. (2000). On Sequential Monte Carlo Sampling Methods for Bayesian Filtering. *Statistics and Computing*, **10**, 197-208.

Gordon, N., Salmond, D. and Smith, A. F. M. (1993). Novel approach to nonlinear and non-Gaussian Bayesian state estimation. *Proc. Inst. Elect. Eng., F*, **140**, 107-113.

Handschin J.E. (1970). Monte Carlo Techniques for Prediction and Filtering of Non-Linear Stochastic Processes. *Automatica* **6**, 555-563.

Handschin J.E. and Mayne D.Q. (1969). Monte Carlo Techniques to Estimate the Conditional Expectation in Multi-stage Non-linear Filtering. *International Journal of Control* **9**, 547-559.

Kitagawa, G. (1996) Monte-Carlo filter and smoother for non-Gaussian nonlinear state space models. *J. Comput. Graph. Statist.* **1**, 1-25.

Kouritzin, M.A. (2015). Microstructure Models with Short-Term Inertia and Stochastic Volatility. *Mathematical Problems in Engineering*, Article ID 323475, in press.

Kouritzin, M.A. and Sun, W. (2005). Rates for Branching Particle Approximations of Continuous-Discrete Filters. *The Annals of Applied Probability* **15**, 2739-2772.

Kouritzin, M.A. and Zeng, Y. (2005). Bayesian Model Selection via Filtering for a Class of Micro-movement Models of Asset Price. *International Journal of Theoretical and Applied Finance* **8**, 97-122.

Kouritzin, M.A. and Zeng, Y. (2005). Weak convergence for a type of conditional expectation: application to the inference for a class of asset price models. *Nonlinear Analysis, Theory, Methods & Applications, Series A* **60** 231-239.

Kurtz, T.G., Xiong, J. (1999). Particle representations for a class of nonlinear SPDEs. *Stochastic Process. Appl.* **83**, 103-126.

Kurtz, T.G. and Xiong, J. (2000). Numerical solutions for a class of SPDEs with application to filtering, Stochastics in Finite and Infinite Dimension: In Honor of Gopinath Kallianpur. Edited by T. Hida, R. Karandikar, H. Kunita, B. Rajput, S. Watanabe and J. Xiong. *Trends in Mathematics*. Birkhauser.

Liu, J.S. and Chen, R. (1998). Sequential Monte-Carlo methods for dynamic systems. *Journal of the American Statistical Association* **93**: 1032-1044.

Vergé, C., Dubarry, C., DelMoral, P. and Moulines, E. (2013). On parallel implementation of sequential Monte Carlo methods: the island particle model. *Statistics and Computing*, **23**: 91-107.

Current address: Department of Mathematical and Statistical Sciences, University of Alberta, Edmonton (Alberta), Canada T6G 2G1

E-mail address: michaelk@ualberta.ca

URL: http://www.math.ualberta.ca/Kouritzin_M.html

http://hdl.handle.net/10402/era.42107 ERA: Education and Research Archive, University of Alberta 21/09/2015