Inference Trees: Adaptive Inference with Exploration

Tom Rainforth[1] **Yuan Zhou**[1] **Xiaoyu Lu**[1] **Yee Whye Teh**[1] **Frank Wood**[1]
Hongseok Yang[2] **Jan-Willem van de Meent**[3]
[1]University of Oxford; [2]KAIST, South Korea; [3]Northeastern University
{rainforth, xiaoyu.lu, y.w.teh}@stats.ox.ac.uk, yuan.zhou@cs.ox.ac.uk,
fwood@robots.ox.ac.uk, hongseok.yang@kaist.ac.kr, j.vandemeent@northeastern.edu

Abstract

We introduce inference trees (ITs), a new adaptive Monte Carlo inference method building on ideas from Monte Carlo tree search. Unlike most existing methods which are implicitly based on pure exploitation, ITs explicitly aim to balance exploration and exploitation in the inference process, alleviating common pathologies and ensuring consistency. More specifically, ITs use bandit strategies to adaptively sample from hierarchical partitions of the parameter space, while simultaneously learning these partitions in an online manner. This allows ITs to "hone-down" on regions of interest, but also maintain uncertainty estimates on whether regions of significant posterior mass have been missed. Though they can be used more generally, we show that ITs are particularly effective when combined with sequential Monte Carlo (SMC), allowing long-range dependencies to be captured and potentially improving beyond what can be achieved by proposal adaptation alone.

1 Introduction

The choice of proposal is one of the key factors in the performance of Monte Carlo methods. It is often difficult, or even impossible, to know what constitutes a good proposal prior to performing inference. For this reason, many methods learn proposals adaptively by utilizing information from previous samples to improve the proposal used at future iterations [3, 7, 8, 11, 14]. However, most methods developed to date do this in a way that is exclusively exploitative – at each iteration they use the best estimate of the proposal so far. This leads to a number of pathologies, often even undermining convergence of the algorithm [5]. To address these issues, we propose developing adaptive methods that explicitly perform exploration – expending computational resources to investigate whether the proposal can be improved, rather than greedily exploiting the best proposal learned so far. To this end, we introduce inference trees (ITs), a new adaptive inference method that builds on ideas from Monte Carlo tree search (MCTS) [2, 19] to control this exploration-exploitation trade-off directly.

ITs partition the target space into disjoint regions, transforming the problem of inference on the full parameter space to a number of smaller inference problems that are combined to an overall estimate, or representation of the posterior, in a manner similar to stratified sampling [6]. The rationale behind this is that it enables one to adaptively allocate computational resources to the different regions using, for example, strategies from the bandit literature [13, 16]. This explicit exploration-exploitation approach has a number of advantages over adaptively learning a single global proposal. Firstly, it breaks the constraint of independent sampling imposed by, for example, importance sampling; allowing explicit allocation computational resources to exploration – i.e. refining our estimate in regions with low estimated posterior mass but high uncertainty. Further, when the aim of inference is to calculate an expectation of a given function, our approach allows one to explicitly aim to reduce the variance of the final estimate, rather than just improving the representation of the posterior.

Another important element of the IT algorithm is that it learns a *hierarchical* partition, i.e. a "tree", in an online manner by building on ideas from MCTS. This results in more fine-grained partitions for regions where the posterior density is large. In essence, we are simultaneously learning a good partitioning structure, establishing the relative frequency to sample each region, and performing inference given these. As such, the IT learning process can be broken down into three key components: traversing the tree to allocate computational resources, refining the tree by running inference at a node or splitting it to expand the tree, and propagating the information gathered from new samples up (and sometimes down) the tree to improve the posterior representation and future traversal strategy.

ITs can be thought of as a meta-inference algorithm that can be combined with existing methods used to perform inference on any given region in the partition. ITs can in principle be formulated using any inference algorithm that provides a consistent estimator of the marginal likelihood (ML). They have the potential to be particularly effective when combined with sequential Monte Carlo (SMC) [4] as they form a means of improving beyond what can be achieved by the so-called one-step optimal proposal. Dealing with long-range dependencies can be challenging in SMC as variables are often fixed before all dependent likelihood terms are incorporated, leading to particle degeneracy.

Naïve strategies for avoiding this tend to be futile – the resampling step always corrects to the intermediate target and thus incorporating lookahead information in proposals is usually frivolous and just reduces the effective sample size. In some cases auxiliary weighting schemes provide a degree of lookahead [10, 15], but these typically entail a substantial increase in computational cost while providing only a short-range lookahead. Moreover, problems with degeneracy can be compounded in the context of adaptation as information is only received for particles that survive the resampling. ITs address these challenges by separately running inference on each partitioned region. This ensures that information is collected for the full space, e.g. through ML estimates, and allows samples from a region to be "forced through" the resampling to deal with long-range dependencies. This can be done without losing the benefits of SMC, by exploiting its effective resource allocation within each region.

2 Partitioning the Target Space

Our aim is to approximate the posterior $p(x_{1:T}|Y)$ of variables $x_{1:T} \in \mathbb{X}_T$, where each $x_t \in \mathcal{X}_t$, given data Y. To do this, we partition $\mathbb{X}_T$ into separate regions $\{A_i\}_{i\in\mathcal{I}}$ (i.e. $\bigcup_{i\in\mathcal{I}} A_i = \mathbb{X}_T$ and $A_i \cap A_j = \emptyset$ for all distinct $i, j \in \mathcal{I}$). We then run inference separately on each region, e.g. using SMC, and combine the separate estimates to an overall estimate. During inference, computational resources are allocated to regions adaptively using bandit methods, which plays a key role in achieving efficiency in solving the overall inference problem.

Effectively partitioning $\mathbb{X}_T$ directly is challenging – typically it will be desirable for the partitions to be neither orthogonal nor even linear. To combat this, we apply a one-to-one mapping of the original state space $\mathbb{X}_T$ into a space $\mathfrak{Z}_T = \bigotimes_{t=1}^{T} \zeta_t$, with each $\zeta_t = [0, 1]$, for which it is hopefully effective to use orthogonal partitions, i.e. partitioning such that each region B_i can be defined using $B_i = \bigotimes_{t=1}^{T} Z_{i,t}$ where each $Z_{i,t} \subseteq \zeta_t$. We assume that each $\mathcal{X}_t \subseteq \mathbb{R}$ and that the mapping constitutes a series of cumulative distribution functions. More precisely, we assume that we have a set of proposals $\{q(x_1), q(x_2|x_1), \ldots, q(x_T|x_{1:T-1})\}$, as would be the case when we are running, for example, SMC inference. We then define our mapping $\mathcal{M} : \mathbb{X}_T \to \mathfrak{Z}_T$ as the collection of the individual mappings $\eta_t : \bigotimes_{u=1}^{t} \mathcal{X}_t \to [0, 1]$ where

$$\eta_t(x_t; x_{1:t-1}) = \int_{-\infty}^{x_t} q_t(x'_t|x_{1:t-1})dx'_t. \tag{1}$$

When $x_{1:T}$ is determined by context, we often use the symbol z_t for $\eta_t(x_t; x_{1:t-1})$. As we show in Appendix B, a valid partitioning of $\mathfrak{Z}_T$ induces a valid partitioning of $\mathbb{X}_T$ and so we can partition $\mathbb{X}_T$ implicitly by partitioning $\mathfrak{Z}_T$ instead. Namely, we define partitions $\{A_i\}_{i\in\mathcal{I}}$ of $\mathbb{X}_T$ using

$$A_i = \{x_{1:T} : (\eta_1(x_1), \eta_2(x_2; x_1), \ldots, \eta_T(x_T; x_{1:T-1})) \in B_i\}.$$

Note the important property that each projection $\eta_t(\cdot; x_{1:t-1})$ maps the distributions $q_t(\cdot|x_{1:t-1})$ to the uniform distribution on the unit interval $\zeta_t = [0, 1]$. That is, the pushforward of each distribution $q_t(\cdot|x_{1:t-1})$ along $\eta_t(\cdot; x_{1:t-1})$ is the uniform distribution on $[0, 1]$. Thus the probability that an $x_{1:T}$ gets mapped to B_i by $\mathcal{M}$ is just the product of the length of $Z_{i,t}$ for every t. We can sample from our proposal truncated to a particular region by uniformly sampling from B_i and applying the inverse mapping. As we are free to choose the form of the proposal, it is possible to ensure this inverse mapping can be calculated analytically by using a proposal with known inverse cumulative distribution function. Furthermore, we can evaluate the density of these truncated proposals using

$$q_{i,t}(x_t|x_{1:t-1}, \{z_t \in Z_{i,t}\}) = \frac{q_t(x_t|x_{1:t-1})\mathbb{I}(z_t \in Z_{i,t})}{\int_{\mathcal{X}_t} q_t(x'_t|x_{1:t-1})\mathbb{I}(z_t \in Z_{i,t})dx'_t} = \frac{q_t(x_t|x_{1:t-1})\mathbb{I}(z_t \in Z_{i,t})}{\int_0^1 \mathbb{I}(z_t \in Z_{i,t})dz_t} \tag{2}$$

where the denominator is just the (known) area of B_i. We can now run inference separately on the partitions using the truncated proposals and later combine the results together (see Appendix A).

3 Tree Formalization

An IT is defined by a set of nodes $\Psi = \{\psi_j\}_{j\in\mathcal{J}\subset\mathbb{N}}$, each of which corresponds to a region B_j (and thus A_j) and takes the form of either a discriminant node or a leaf node. Discriminant nodes

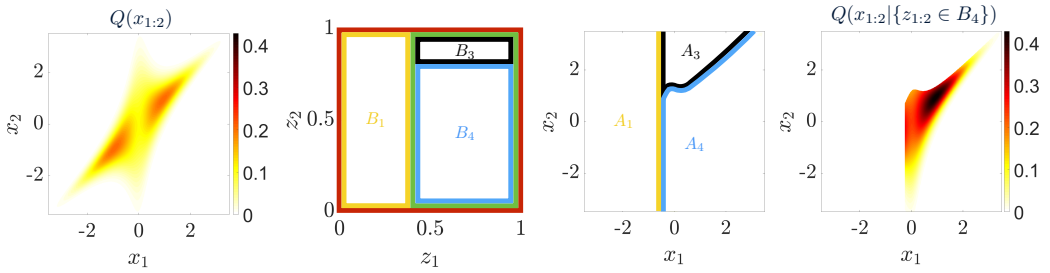


Figure 1: Representation of an IT with proposal $Q(x_{1:2}) = \mathcal{N}(x_1; 0, 1)\mathcal{N}\left(x_2; x_1, \sqrt{\frac{1}{0.5+4x_1^2}}\right)$ [far left]. [Second left] the hierarchical partitioning of $\mathfrak{Z}_T$ imposed by the tree where nodes $1, 3$, and 4 are leaves; $B_2 = B_3 \cup B_4$ and $B_0 = B_1 \cup B_2$; and $\alpha_0 = 1$, $s_0 = 0.4$, $\alpha_2 = 2$, $s_2 = 0.8$. [Second right] the partitioning implied by $Q(x_{1:2})$ and the leaf nodes on the target space $\mathbb{X}_T$, where we note that the partition between A_3 and A_4 is nonlinear. [Far right] the proposal truncated to A_4 and renormalized.

have child nodes, corresponding to sub-partitions, but leaf nodes do not. The former is defined by $(l_j, r_j, \alpha_j, s_j, c_j, \hat{\varrho}_j)$ and the latter by $(c_j, \hat{\varrho}_j)$. Here $l_j, r_j \in \mathcal{J}$ are the indices of the child nodes, $\alpha_j \in \{1, \ldots, T\}$ is the index of the variable z_{α_j} used to the split in the cumulative density space, $s_j \in [0, 1]$ is the point of the split, c_j is a node weight, and $\hat{\varrho}_j$ is an estimate (or collection of estimates) produced by running inference at that node. We assume that $\hat{\varrho}_j$ can be estimated in an online fashion, using N_j to denote the budget it has been assigned thus far. For SMC, N_j is a number of sweeps. Through this definition, a tree induces a nested orthogonal partitioning of $\mathfrak{Z}_T$ (see Figure 1) where every node is associated with a subset $B_j \subseteq [0, 1]^T$. The root node is assigned the entire space, $B_0 = [0, 1]^T$, and each discriminant node splits its region between its children:

$$B_{l_j} = B_j \cap \{z_{1:T} \in [0, 1]^T \mid z_{\alpha_j} \leq s_j\}, \quad B_{r_j} = B_j \cap \{z_{1:T} \in [0, 1]^T \mid z_{\alpha_j} > s_j\}. \tag{3}$$

As a result, the B_j's for the leaf nodes form a partition of $[0, 1]^T$. Further, if $\mathcal{J}_d \subseteq \mathcal{J}$ is the subset of nodes that are at depth d or are leaf nodes whose depth is less than d, then $\bigcup_{j \in \mathcal{J}_d} B_j = [0, 1]^T$.

Using the fact that the each region B_j of a discriminant node j is the union of those of its children, we define an IT estimator recursively, starting at the leaves and working upwards to the root node. Specifically, we define an estimate for the partition B_j as $\hat{\mu}_j = \hat{\varrho}_j$ if j is a leaf node and

$$\hat{\mu}_j = \frac{c_j - \left(c_{l_j} + c_{r_j}\right)}{c_j} \hat{\varrho}_j + \frac{c_{l_j} + c_{r_j}}{c_j} \left(\hat{\mu}_{l_j} + \hat{\mu}_{r_j}\right) \tag{4}$$

otherwise. Thus $\hat{\mu}_j$ is a weighted combination of a local estimate from the node ($\hat{\varrho}_j$) and an estimate computed from its descendants recursively. Our approach is backed up by the following consistency result in the number of IT iterations K, proof and further details for which are given in Appendix B.

Theorem 1. *If each $\hat{\varrho}_j$ in an IT converges weakly as $N_j \to \infty$ and if Assumptions 1, 2 and 3 hold, then all estimates calculated using an IT also converge weakly as $K \to \infty$.*

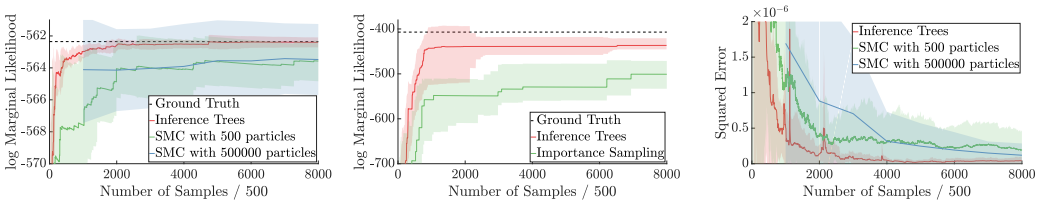
Consequently, subject to some weak assumptions, consistency is achieved regardless of the strategy for choosing c_j and N_j, i.e. regardless of our resource allocation strategy. Note, though, that the estimates produced by ITs are biased, even when the base estimation scheme is not. This is because the N_j are random variables that are correlated with the estimate such that $\mathbb{E}[\hat{\varrho}_j \mid N_j = n_1] \neq \mathbb{E}[\hat{\varrho}_j \mid N_j = n_2]$ for $n_1 \neq n_2$ in general – a lower value of N_j will typically indicate a lower ML.

4 Tree Learning Algorithm

The process of constructing an IT starts with a single root node and then iterates traversal, refinement, and propagation steps, forming a continuously improving online estimation scheme. We have shown that ITs lead to consistent estimates regardless of the approach taken for this, provided some weak assumptions are satisfied. However, their practical performance will inevitably heavily depend on this learning algorithm. Due to space restrictions, we provide only key high-level ideas of our approach here and refer the reader to Appendix C for a more complete description.

4.1 Propagation

Whenever we run inference at a node, we need to propagate that information through the tree to refine our overall estimate and improve the future traversal. The main way this is done is recursively applying (4) to propagate the new estimates *up* through the tree – whenever we update a node we also update all its ancestors in the same manner as MCTS. However, unlike in MCTS, ITs sometimes



(a) Chaotic Dynamics (b) GMM (c) Network Tail Integral

Figure 2: Convergence of ITs (with 500 particles for each evaluation) compared to base inference algorithms, solid lines show mean over 10 runs, with shaded region showing $\pm$ one standard deviation.

run inference at intermediate nodes. The main rationale for doing this is that running inference at an intermediate node and propagating the resultant *down* through the tree can be a highly effective means of exploration. The $\hat{p}_j$ of the child nodes are not updated in this manner, but a separate ancestral estimate is calculated that is incorporated into the traversal strategy.

4.2 Traversal

Our procedure for allocating computational resources to nodes follows a similar method to upper confidence trees [12], starting at the root note and recursively make traversal decisions based on upper confidence bounding. However, we introduce a number of significant differences such as an option to stop at the current node rather than recursing to one of the children, using explicit uncertainty estimates to better guide the exploration, annealing the target to reflect the heavy-tailed nature of most estimates, and the use of alternating parameter setups to accelerate the learning process by allowing the tree to get rapidly deeper in certain areas. As the goal of posterior inference is often to estimate an expectation of some function, say f, we adapt our traversal strategy to the "unknown f" and "known f" cases, using the ML and the standard error of the expectation estimate respectively as our "reward" for the each. The optimality of the latter was shown by [6] in the non-hierarchical setting.

4.3 Refinement

When a node is chosen by the traversal strategy, there are two ways we can refine the tree. We can just update the local estimate or, if the node is a leaf node, we can split that node to expand the tree. The two considerations here are whether to split and how to split. The former is controlled by only attempting to split once N_j reaches a certain threshold and only accepting a split if it passes a usefulness test (namely a significance test on the partition estimates originating from different distributions) – we generally wish to avoid unnecessary over-splitting. For the latter, we use the samples generated from the current node to estimate where the best point to split will be using entropy metric that encourages splits which can be best exploited later by the traversal strategy.

5 Experiments

We present preliminary results for three models, details for which are given in Appendix D. Two of these are"unknown f" problems, for which our aim is approximation of the posterior. Namely, we consider the chaotic dynamical system problem introduced by [18] and a Gaussian mixture model (GMM). In both cases, ITs are used to control the inference of selected parameters (respectively the dynamics parameters and mixture component means), with SMC and importance sampling used as the base inference algorithms respectively. Comparisons to using the base inference algorithms in isolation, measured in terms of the ML estimate, are given in Figures 2(a) and 2(b), which demonstrate noticeable improvements. For the chaos problem, we further see improvements compared to using 1000 times more particles (with the number of iterations reduced respectively). Our third model is a "known f" problem where we estimate a tail integral in a network model. Here ITs control all the parameters and we see improvements in the mean squared error over using the base inference algorithm (SMC) in isolation, including again the case where we use more particles per sweep.

Though substantial improvements are observed for the GMM, it is still noticeable that the ML estimates are far from perfect. Closer inspection shows that ITs have not always managed to "find" the largest mode. We believe the reason for this stems from the fact that although SMC and importance sampling provide unbiased ML estimates, it is typically far more probable they will underestimate the ML than overestimate it. Consequently, naïve uncertainty estimates usually fail to capture the probability that a region makes a significant contribution to the overall expectation and therefore regions can be missed by the traversal strategy. Work is ongoing to develop more robust uncertainty estimates using extreme value theory and to share information between nodes that are close in the target space but are in different branches of the tree in order to help identify missing regions.

References

[1] C. Andrieu and G. O. Roberts. The Pseudo-marginal Approach for Efficient Monte Carlo Computations. *The Annals of Statistics*, pages 697–725, 2009.

[2] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton. A Survey of Monte Carlo Tree Search Methods. *IEEE Transactions on Computational Intelligence and AI in games*, 4(1):1–43, 2012.

[3] O. Cappé, A. Guillin, J.-M. Marin, and C. P. Robert. Population Monte Carlo. *Journal of Computational and Graphical Statistics*, 13(4):907–929, 2004.

[4] O. Cappé, S. J. Godsill, and E. Moulines. An Overview of Existing Methods and Recent Advances in Sequential Monte Carlo. *Proceedings of the IEEE*, 95(5):899–924, 2007.

[5] O. Cappé, R. Douc, A. Guillin, J.-M. Marin, and C. P. Robert. Adaptive Importance Sampling in General Mixture Classes. *Statistics and Computing*, 18(4):447–459, 2008.

[6] A. Carpentier, R. Munos, and A. Antos. Adaptive Strategy for Stratified Monte Carlo Sampling. *Journal of Machine Learning Research*, 16:2231–2271, 2015.

[7] J. Cornebise, É. Moulines, and J. Olsson. Adaptive Methods for Sequential Importance Sampling with Application to State Space Models. *Statistics and Computing*, 18(4):461–480, 2008.

[8] J. Cornuet, J.-M. MARIN, A. Mira, and C. P. Robert. Adaptive Multiple Importance Sampling. *Scandinavian Journal of Statistics*, 39(4):798–812, 2012.

[9] R. Douc, A. Guillin, J.-M. Marin, and C. P. Robert. Convergence of Adaptive Mixtures of Importance Sampling Schemes. *The Annals of Statistics*, pages 420–448, 2007.

[10] A. Doucet, M. Briers, and S. Sénécal. Efficient Block Sampling Strategies for Sequential Monte Carlo Methods. *Journal of Computational and Graphical Statistics*, 15(3):693–711, 2006.

[11] S. Gu, Z. Ghahramani, and R. E. Turner. Neural Adaptive Sequential Monte Carlo. In *Advances in Neural Information Processing Systems*, pages 2629–2637, 2015.

[12] L. Kocsis and C. Szepesvári. Bandit Based Monte-Carlo Planning. In *ECML*, volume 6, pages 282–293. Springer, 2006.

[13] T. L. Lai and H. Robbins. Asymptotically Efficient Adaptive Allocation Rules. *Advances in applied mathematics*, 6(1):4–22, 1985.

[14] F. Liang, C. Liu, and R. Carroll. *Advanced Markov Chain Monte Carlo Methods: Learning from Past Samples*, volume 714. John Wiley & Sons, 2011.

[15] M. Lin, R. Chen, J. S. Liu, et al. Lookahead Strategies for Sequential Monte Carlo. *Statistical Science*, 28(1):69–94, 2013.

[16] J. Neufeld, A. Gyorgy, C. Szepesvari, and D. Schuurmans. Adaptive Monte Carlo via Bandit Allocation. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32, pages 1944–1952, 2014.

[17] A. B. Owen. *Monte Carlo theory, methods and examples*. 2013.

[18] T. Rainforth, T. A. Le, J.-W. van de Meent, M. A. Osborne, and F. Wood. Bayesian Optimization for Probabilistic Programs. In *Advances in Neural Information Processing Systems*, pages 280–288, 2016.

[19] G. Tesauro, V. Rajan, and R. Segal. Bayesian inference in Monte-Carlo tree search. *arXiv preprint arXiv:1203.3519*, 2012.

A Combining Estimates from Partitions

We now consider how estimates can be combined from different partitions to form an overall estimate. In the interest of notational simplicity, we will first consider the case of using importance sampling for the running inference locally. Let

$$Q\left(x_{1:T} \mid \{z_{1:T} \in B_i\}\right) = q_{i,1}(x_1|\{z_1 \in Z_{i,1}\}) \prod_{t=2}^{T} q_{i,t}(x_t|x_{1:t-1}, \{z_t \in Z_{i,t}\}) \tag{5}$$

be the overall truncated proposal for region B_i. We now have for marginal likelihood estimation

$$\begin{aligned}
p(Y) &= \int p(x_{1:T}, Y)\, dx_{1:T} = \int \sum_{i \in \mathcal{I}} \mathbb{I}(x_{1:T} \in A_i) p(x_{1:T}, Y)\, dx_{1:T} \\
&= \sum_{i \in \mathcal{I}} \int \mathbb{I}(z_{1:T} \in B_i) p(x_{1:T}, Y)\, dx_{1:T} = \sum_{i \in \mathcal{I}} \int p(x_{1:T}, Y) \prod_{t=1}^{T} \mathbb{I}(z_t \in Z_{i,t})\, dx_{1:T} \\
&= \sum_{i \in \mathcal{I}} \mathbb{E}_{Q(x_{1:T}|\{z_{1:T} \in B_i\})} \left[\frac{p(x_{1:T}, Y) \prod_{t=1}^{T} \mathbb{I}(z_t \in Z_{i,t})}{Q\left(x_{1:T} \mid \{z_{1:T} \in B_i\}\right)} \right] \\
&\approx \sum_{i \in \mathcal{I}} \hat{\upsilon}_i \quad \text{where each} \quad \hat{\upsilon}_i = \frac{1}{N_i} \sum_{n=1}^{N_i} w_n^i, \quad w_n^i = \frac{p(\hat{x}_{1:T,i}^n, Y) \prod_{t=1}^{T} \mathbb{I}(\hat{z}_{t,i}^n \in Z_{i,t})}{Q\left(\hat{x}_{1:T,i}^n \middle| \{\hat{z}_{1:T,i}^n \in B_i\}\right)},
\end{aligned} \tag{6}$$

$\hat{x}_{1:T,i}^n \sim Q\left(x_{1:T} \mid \{z_{1:T} \in B_i\}\right)$, and $\hat{z}_{1:T,i}^n$ is implied by $\hat{x}_{1:T,i}^n$. Note here that each $\hat{\upsilon}_i$ is an estimate of the marginal probability $p(\{z_{1:T} \in B_i\}, Y) = p(\{x_{1:T} \in A_i\}, Y)$.

If we instead consider the case of calculating an expectation we now have

$$\begin{aligned}
\mathbb{E}_{p(x_{1:T}|Y)}[f(x_{1:T})] &= \frac{1}{p(Y)} \int p(x_{1:T}, Y) f(x_{1:T})\, dx_{1:T} \\
&= \frac{1}{p(Y)} \sum_{i \in \mathcal{I}} \int p(x_{1:T}, Y) f(x_{1:T}) \prod_{t=1}^{T} \mathbb{I}(z_t \in Z_{i,t})\, dx_{1:T} \\
&= \frac{1}{p(Y)} \sum_{i \in \mathcal{I}} \mathbb{E}_{Q(x_{1:T}|\{z_{1:T} \in B_i\})} \left[\frac{p(x_{1:T}, Y) f(x_{1:T}) \prod_{t=1}^{T} \mathbb{I}(z_t \in Z_{i,t})}{Q\left(x_{1:T} \mid \{z_{1:T} \in B_i\}\right)} \right] \\
&\approx \frac{\sum_{i \in \mathcal{I}} \hat{\varrho}_i}{\sum_{i \in \mathcal{I}} \hat{\upsilon}_i} \quad \text{where} \quad \hat{\varrho}_i = \frac{1}{N_i} \sum_{n=1}^{N_i} w_n^i f\left(\hat{x}_{1:T,i}^n\right),
\end{aligned} \tag{7}$$

each $\hat{\varrho}_i$ is an estimate of $\mu_i := \mathbb{E}_{p(x_{1:T}|Y)}[f(x_{1:T}) p(Y) \mathbb{I}(x_{1:T} \in A_i)]$ and the weights are defined as before. Note that we are using self-normalization for the importance sampling, but that we are doing this globally rather than locally – though the w_n^i are not normalized, the normalization constant for all weights is the same global constant $p(Y)$ (because the proposals are correctly normalized within their target area). In general, we will propagate *unnormalized* estimates $\hat{\varrho}_i$, only normalizing them at the root node using the marginal probability estimates $\hat{\upsilon}_i$ which are themselves propagated through the tree separately. Note that $f\left(x_{1:T}\right) = 1,\ \forall x_{1:T} \in \mathbb{X}_T$ is a special case giving marginal likelihood estimates, so we can use the same approaches for $\hat{\upsilon}_i$ as for $\hat{\varrho}_i$. Namely, analogously to $\hat{\mu}_j$, we define $\hat{\omega}_j = \hat{\upsilon}_i$ for the leaf nodes and for the others

$$\hat{\omega}_j = \frac{c_j - \left(c_{l_j} + c_{r_j}\right)}{c_j} \hat{\upsilon}_j + \frac{c_{l_j} + c_{r_j}}{c_j} \left(\hat{\omega}_{l_j} + \hat{\omega}_{r_j}\right). \tag{8}$$

We can calculate a Monte Carlo empirical measure approximation for the posterior using the same approach as $\hat{\varrho}_j$, namely

$$p(x_{1:T}|Y) \approx \frac{\sum_{i \in \mathcal{I}} \frac{1}{N_i} \sum_{n=1}^{N_i} w_n^i \delta_{\hat{x}_{1:T,i}^n}(x_{1:T})}{\sum_{i \in \mathcal{I}} \hat{\upsilon}_i}. \tag{9}$$

Note that we are combining estimates from different partitions in an unweighted manner – there is no need to apply correction factors for the strategy used to assign computational resources. We can do this because the weights are calculated using correctly normalized proposal distributions as per (2).

Using SMC inference at each of the nodes follows the same derivation with the weights appropriately redefined (namely they are the product of the marginal likelihood estimate and the local self-normalized weight from the sweep). For notational consistency later in the tree training algorithm, will use n to index an SMC sweep, rather than individual sample, such that each $\hat{x}^n_{1:T,i}$ and w^i_n itself corresponds to a set of particles and corresponding weights.

More generally, we can use any method for which each $\hat{\varrho}_i$ and $\hat{v}_i$ are consistent estimates as $N_i \to \infty$. An important consequence of this is that we can use ITs in pseudo-marginal [1] situations when the likelihood cannot be directly evaluated but can be estimated unbiasedly. This means that, for example, we can run SMC where there as some latent variables not directly control by the IT algorithm, i.e. $p(x_{1:T}|Y) = \int p(x_{1:T}, u_{1:S}|Y)du_{1:S}$ and we run SMC inference on $p(x_{1:T}, u_{1:S}|Y)$.

These combination methods are strictly for combining disjoint partitions. We can thus use them to combined estimates from the left and right children of discriminant nodes as per the second term in (4). To combine estimates from parents and their children, we simply note that the weighted sum of two consistent independent consistent estimators is itself consistent if the weights sum to 1. Thus we can view $\frac{c_j - (c_{l_j} + c_{r_j})}{c_j}$ as a parent weight and $\frac{c_{l_j} + c_{r_j}}{c_j}$ as a child weight, which are then used to combine the two independent estimators in (4).

B Theoretical Justification

In this section, we will show the correctness of our strategy for splitting a target space, the consistency of combing estimators, and the convergence of our algorithm.

We first demonstrate that our partition of $\mathfrak{Z}_T$ leads to valid partitions on $\mathbb{X}_T$, i.e. that the A_i are disjoint and the union of all the A_i is the full input space.

Lemma 1. *Let $\mathbb{X}_T = \bigotimes_{t=1}^T \mathcal{X}_t$ be the original target space and $\mathfrak{Z}_T = \bigotimes_{z=1}^T \zeta_t$ the mapping of this target space produced by a sequential application of the collection of functions $\{\eta_t\}_{1\leq t\leq T}$ where each η_t is a one-to-one map from $\mathcal{X}_t$ to ζ_t parameterized by $x_{1:t-1} \in \bigotimes_{u=1}^{t-1} \mathcal{X}_u$. Then, every region $\{B_i\}_{i\in\mathcal{I}}$ of $\mathfrak{Z}_T$ induces a region $\{A_i\}_{i\in\mathcal{I}}$ of $\mathbb{X}_T$ defined as follows:*

$$A_i = \{x_{1:T} \mid (\eta_1(x_1), \ldots, \eta_t(x_t; x_{1:t-1}), \ldots, \eta_T(x_T; x_{1:T-1})) \in B_i\}.$$

Furthermore, if $\bigcup_{i=1}^{\mathcal{I}} B_i = \mathfrak{Z}_T$ and $B_i \bigcap B_j = \emptyset$ for any $i \neq j$, then it follows that $\bigcup_{i=1}^{\mathcal{I}} A_i = \mathbb{X}_T$ and $A_i \bigcap A_j = \emptyset$ for any $i \neq j$.

Proof. Let $\mathcal{M}$ be the following function from $\mathbb{X}_T$ to $\mathfrak{Z}_T$:

$$\mathcal{M}(x_{1:T}) = (\eta_1(x_1), \ldots, \eta_t(x_t; x_{1:t-1}), \ldots, \eta_T(x_T; x_{1:T-1})).$$

Then, $\{A_i\}_{i\in\mathcal{I}}$ is by definition the family of the inverse images of $\{B_i\}_{i\in\mathcal{I}}$ by $\mathcal{M}$. It is straightforward follows by induction that $\mathcal{M}$ is injective (i.e. a one-to-one mapping from $\mathbb{X}_T$ to $\mathfrak{Z}_T$), while, by definition of $\mathfrak{Z}_T$, $\mathcal{M}$ is also surjective (i.e. there no elements in $\mathfrak{Z}_T$ without a preimage in $\mathbb{X}$). Because $\mathcal{M}$ is injective and surjective, each $z_{1:T} \in \mathfrak{Z}_T$ must correspond to exactly one unique $x_{1:T} \in \mathbb{X}_T$ and vice-versa. It therefore immediately follows that each $x_{1:T} \in \mathbb{X}_T$ must be mapped to a unique B_i and each $z_{1:T} \in \mathfrak{Z}_T$ must have a preimage in a unique A_i. Consequently, disjoint B_i imply disjoint A_i and $\bigcup_{i=1}^{\mathcal{I}} B_i = \mathfrak{Z}_T$ implies $\bigcup_{i=1}^{\mathcal{I}} A_i = \mathbb{X}_T$ as required. □

We next demonstrate that for any partitioning $\{B_i\}_{i\in\mathcal{I}}$ and set of consistent estimators for each partition, then the combination strategies given in Appendix A similarly lead to consistent estimators. Moreover, we demonstrate that this convergence holds when we combine multiple sets of estimators, each with their own partitioning, for example the parent estimator and children estimator in (10). Proving the convergence of the IT algorithm will be straightforward given this Lemma. At a high-level we make three assumptions: each constituent estimator is consistent in isolation, each set of estimators only has finite combination weight in the limit of large overall computational budget if each of constituent region estimators receives a finite proportion of that overall computational budget, and the number of each regions is finite for each estimator set. For formally we have

Assumption 1. *For every independent estimator set $\ell \in \{1, \ldots, L\}$, we are given a projection $\eta_\ell : \mathbb{X}_T \to \mathfrak{Z}_T$, a partition $\{B_{\ell,i}\}_{i\in\mathcal{I}_\ell}$ of the cumulative density space $\mathfrak{Z}_T$, and a family $\{\hat{p}^{N_{\ell,i}}_{\ell,i}\}_{i\in\mathcal{I}_\ell}$ of estimated measures on $\mathbb{X}_T$ for all $N \geq 1$:*

$$\hat{p}^{N_{\ell,i}}_{\ell,i}(\cdot) = \frac{1}{N_{\ell,i}} \sum_{n=1}^{N_{\ell,i}} w^n_{\ell,i}\, \delta_{\hat{x}^n_{1:T,\ell,i}}(\cdot)$$

for some random variables $w_{\ell,i}^n$ and $\hat{x}_{1:T,\ell,i}^n$ such that each $\hat{p}_{\ell,i}^{N_{\ell,i}}$ converges weakly to the following measure on $\mathbb{X}_T$ as $N_{\ell,i} \to \infty$

$$p(x_{1:T}, Y)\mathbb{I}(\eta_\ell(x_{1:T}) \in B_{\ell,i}).$$

Assumption 2. *Let $k_\ell : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ be combination weight functions which produce unnormalized combination weights $k_\ell(N_\ell)$ when provided with the total number of samples used for the corresponding estimator set $N_\ell = \sum_{i \in \mathcal{I}_\ell} N_{\ell,i}$ such that $\lim_{N_\ell \to \infty} k_\ell(N_\ell) = \infty$ for each ℓ, each $k_\ell(N_\ell)$ is finite for any finite N_ℓ, and $\sum_{\ell=1}^L k_\ell(N_\ell) > 0$ whenever $K = \sum_{\ell=1}^L N_\ell > 0$. We further assume that for each estimator set ℓ, either all of the $N_{\ell,i}$ tend to infinity or none of them. More precisely, we assume there is a non-empty subset $\mathcal{L}_0 \subseteq \{1, \ldots, L\}$ such that for all $\ell \in \mathcal{L}_0$ and $i \in \mathcal{I}_\ell$,*

$$\lim_{K \to \infty} N_{\ell,i} = \infty$$

and for all $\ell \notin \mathcal{L}_0$,

$$\lim_{K \to \infty} N_\ell < \infty$$

almost surely.

Assumption 3. *Each $\mathcal{I}_\ell$ is a finite set. In the context of ITs, this is equivalent to assuming that the depth of the tree remains bounded.*

The last of these assumptions can probably be relaxed to $\mathcal{I}_\ell$ being a countable set, but as it will be algorithmically beneficial to ensure that the depth of the tree remains bounded, this case is of little interest anyway. The need for the second assumption is to ensure that any individual estimator which only has finite computational budget in the limit of large overall budget is given zero weight after normalization.

We are now ready to demonstrate the consistency of our estimator combination.

Lemma 2. *If Assumptions 1, 2 and 3 hold, then*

$$\hat{p}^{\{N_{\ell,i}\}_{\ell,i}} = \frac{1}{\sum_{\ell=1}^L k_\ell(N_\ell)} \sum_{\ell=1}^L k_\ell(N_\ell) \sum_{i \in \mathcal{I}_\ell} \hat{p}_{\ell,i}^{N_{\ell,i}} \tag{10}$$

converges weakly to the measure $p(x_{1:T}, Y)$ on $\mathbb{X}_T$ as the $N_{\ell,i}$ tend to ∞.

Proof. By assumption we have that each $\hat{p}_{\ell,i}^{N_{\ell,i}}$ converges weakly to the measure

$$p(x_{1:T}, Y)\mathbb{I}(\eta_\ell(x_{1:T}) \in B_{\ell,i})$$

as $\hat{p}_{\ell,i}^{N_{\ell,i}}$ tends to ∞. Thus, for each $\ell \in \mathcal{L}_0$,

$$\sum_{i \in \mathcal{I}_\ell} \hat{p}_{\ell,i}^{N_{\ell,i}}$$

converges weakly to the measure

$$\sum_{i \in \mathcal{I}_\ell} p(x_{1:T}, Y)\mathbb{I}(\eta_\ell(x_{1:T}) \in B_{\ell,i}) = p(x_{1:T}, Y)$$

as the $K \to \infty$. The estimates for $\ell \notin \mathcal{L}_0$ need not converge but do not affect the final estimate as

$$\lim_{K \to \infty} \frac{k_\ell(N_\ell)}{\sum_{\ell=1}^L k_\ell(N_\ell)} = 0 \quad \forall \ell \notin \mathcal{L}_0.$$

To show the claim of this theorem, we now consider an arbitrary bounded continuous function $f : \mathbb{X}_T \to \mathbb{R}$ for which we have

$$\begin{aligned}
\int f(x_{1:T})\, \hat{p}^{\{N_{\ell,i}\}_{\ell,i}}(dx_{1:T}) &= \int f(x_{1:T}) \times \frac{1}{\sum_{\ell=1}^L k_\ell(N_\ell)} \sum_{\ell=1}^L k_\ell(N_\ell) \sum_{i \in \mathcal{I}_\ell} \hat{p}_{\ell,i}^{N_{\ell,i}}(dx_{1:T}) \\
&= \frac{1}{\sum_{\ell=1}^L k_\ell(N_\ell)} \sum_{\ell=1}^L k_\ell(N_\ell) \sum_{i \in \mathcal{I}_\ell} \int f(x_{1:T}) \hat{p}_{\ell,i}^{N_{\ell,i}}(dx_{1:T})
\end{aligned}$$

which using Assumptions 1 and 2 converges as $K \to \infty$ to

$$\begin{aligned}
&\frac{1}{\sum_{\ell \in \mathcal{L}_0} k_\ell\left(N_\ell\right)} \sum_{\ell \in \mathcal{L}_0} k_\ell\left(N_\ell\right) \sum_{i \in \mathcal{I}_\ell} \int f(x_{1:T}) \mathbb{I}(\eta_\ell(x_{1:T}) \in B_{\ell,i})\, p(dx_{1:T}, Y) \\
&\qquad = \frac{1}{\sum_{\ell \in \mathcal{L}_0} k_\ell\left(N_\ell\right)} \sum_{\ell \in \mathcal{L}_0} k_\ell\left(N_\ell\right) \int f(x_{1:T})\, p(dx_{1:T}, Y) \\
&\qquad = \int f(x_{1:T})\, p(dx_{1:T}, Y)
\end{aligned}$$

where we have implicitly used the result of Lemma 1. □

This result firstly conveys that if we combine convergent estimators for the partitioned parts of the overall target, we get a convergent estimator for the target. Secondly, it implies that we can similarly combine a number of estimates for the target, which come from different partitionings. For example, we can combine a posterior estimate $\hat{p}(x_{1:T}, Y, \{z_{1:T} \in B\})$ for the trivial partition $\{B\}$ of B, with that given by combining $\hat{p}(x_{1:T}, Y, \{z_{1:T} \in B_\ell\})$ and $\hat{p}(x_{1:T}, Y, \{z_{1:T} \in B_r\})$ for the partitioned parts B_ℓ and B_r where $B = B_\ell \cup B_r$, in a manner that preserves consistency. These results hold independently of how the k_ℓ are chosen, provided Assumption 2 holds. However, the variances of the associated estimates are likely to depend heavily on the choice of k_ℓ – we wish to place more weight on the partitionings with lower variance estimates.

A critical point is that the combination of estimators does not require any correction factor for the number of times that an estimator and a partition were "proposed" – i.e. we do not need to correct for the fact that more computational resources are provided for some estimates than others or because some partitions of the space are potentially larger than others. All such potential factors either cancel out, or are dealt with by the correct normalization of the truncated proposal. As such, any strategy on deciding the partitions or how often a partition is proposed only need satisfy the stated assumptions to ensure consistency. We are now thus ready to prove Theorem 1 from the main paper as follows.

Proof for Theorem 1. The result follows from Lemma 2 and induction through the tree. Specifically, we first note that Assumption 3 guarantees that the number of leaf nodes in the tree is bounded. It further directly follows from Assumptions 1 and 2 and Lemma 2 that the estimates $\hat{\mu}_j$ for the leaf nodes and their immediate parent nodes (calculated using (4)) all either converge (because they are assigned infinite budget) or do not affect the final estimate (because their combination weights become dominated by an ancestor in the tree). By induction and repeated application of Lemma 2, the estimates at all the nodes in the tree calculated recursively using (4) also converge or do not affect the final estimate. Because the root node has no ancestors, it cannot fall into the latter category and so converges to the full target problem as required. Note that by choosing node weight functions c_j that locally satisfy the same requirements as given combination weights k_ℓ in the non-hierarchal case, this trivially induces valid estimate combination weights as required. □

C Additional Algorithm Details

In this section we provide additional details on the traversal and refinement strategies of the IT algorithm. Developing these approaches is the subject of ongoing work and there are a plethora of approaches one might envisage using under our general IT framework, providing numerous opportunities for extensions.

C.1 Traversal

The aim of the traversal process is to allocate computational resources optimally through the tree. We, therefore, start by considering what it means to allocate resources optimally, noting that the hierarchical nature of the tree means we only need to consider the optimal policy for the respective frequency of sampling from the left and right child nodes, or stopping at the traversal at the parent.

Typically the target for an inference scheme is to calculate the expectation of some target function f under the posterior distribution $p(x_{1:T}|Y)$, namely $\mathbb{E}_{p(x_{1:T}|Y)}\left[f(x_{1:T})\right] = \mu/p(Y)$ where $\mu = \mathbb{E}_{p(x_{1:T}|Y)}\left[f(x_{1:T})p(Y)\right]$ and our slightly unusual notation is used to highlight that estimates for μ and $p(Y)$ are propagated separately. A common example would be in Bayesian model averaging for the posterior predictive distribution. Most adaptive sample schemes look to only to approximate the posterior in the most accurate way, ignoring the influence of the f on the estimation. Clearly, this is

inferior when f is known, as it ignores the fact that f may have higher variability in some regions than others, such that the accuracy in those regions is more impactful on the error in the overall estimate. Inference trees are capable of operating in either a way that is aware of a target function f or not. These different cases lend themselves to different traversal strategies which we now consider separately. We will use the notation P_ℓ to indicate the probability of choosing the left node in the traversal throughout, with the probability of choosing the right node $P_r = 1 - P_\ell$. We will similarly drop the j index notation when it is clear from the context what the index of the parent node is.

C.1.1 Unknown f

When f is unknown, ITs aim to provide the most accurate representation of the posterior $p(x_{1:T}|Y)$ as this is well known to provide the optimal worst case error [17]. ITs and thus aim to sample children in proportion to their marginal probabilities

$$p(\{z_{1:T} \in B_j\}, Y) = p(\{x_{1:T} \in A_j\}, Y) \propto p(\{x_{1:T} \in A_j\}|Y)$$

and so we can think of

$$P_\ell^* = \frac{p(\{z_{1:T} \in B_\ell\}, Y)}{p(\{z_{1:T} \in B_\ell\}, Y) + p(\{z_{1:T} \in B_r\}, Y)}$$

as being the optimal traversal for the relative frequency of choosing each child. The traversal strategy of ITs for an unknown f consequently looks to use the empirical estimates of $p(\{z_{1:T} \in B_j\}, Y)$, namely $\hat{\omega}_j$, to tune how relatively often different paths are traversed. A naïve possible traversal strategy would thus be to simply sample a child node in proportion to its empirical estimate, i.e. $P_\ell = \frac{\hat{\omega}_\ell}{\hat{\omega}_\ell + \hat{\omega}_r}$. This strategy could be viewed as using a proposal for sampling the child node that is the empirical distribution of the samples thus far, noting that this relates to many existing adaptive sampling methods [7, 11, 5, 9] in that it minimizes the Kullback-Leibler divergence from the empirical sample distribution to the proposal.

The problem with such a strategy is that it does not take into account that the utility from sampling a child node originates not only in the improvement of the overall estimator, but also from improving the estimates for $\hat{\omega}_\ell$ and $\hat{\omega}_r$ to improve the future traversal strategy, e.g. by bringing P_ℓ closer to its optimal value P_ℓ^*. In other words, the samples we take now will help us sample better in the future and so we need to not only exploit what we currently believe are good proposal distributions, but also explore the space of proposals. In particular, it is desirable to account for the fact that the samples generated thus far might not be representative and so a proposal substantially different to any tried before might be needed to generate good samples. The lack of consideration for such an explorational component to adaptation may account for the pathologies and sometimes disastrous performance of some adaptive importance sampling schemes, see e.g. [5]. Such schemes could be seen as greedy, in the sense that the proposal chosen is always that which best matches the estimate so far. As an extreme but realistic example of what might happen if such as scheme was used for ITs, consider the case were $p(x_{1:T}, \{z_{1:T} \in B_\ell\}, Y) = 0$ for certain values of $x_{1:T}$. It is thus possible that initial estimate $\hat{\omega}_\ell = 0$ even if $p(\{z_{1:T} \in B_\ell\}, Y) \neq 0$. In this scenario, presuming $\hat{\omega}_r \neq 0$, then P_ℓ would be set to 0 and remain so indefinitely if taking the naïve strategy $P_\ell = \frac{\hat{\omega}_\ell}{\hat{\omega}_\ell + \hat{\omega}_r}$. Such an approach would violate Assumption 2 as it would allow the right child to be traversed infinitely often, without traversing the left infinitely often.

To avoid such behavior, we introduce two innovations. Firstly we note that it is desirable to sample paths were the variance of the marginal probability estimate is high. To propagate errors up the tree, we use the fact that the estimator variances are additive because the the estimators are sampled independently.[1] Specifically we have for the general f case (again exploiting that $\hat{\omega}_j$ is a particular case of $\hat{\mu}_j$)

$$\begin{aligned}
\sigma_j^2 = \mathbb{E}\left[(\hat{\mu}_j - \mu_j)^2\right] &= \mathbb{E}\left[\left(\frac{c_j - c_\ell - c_r}{c_j}\hat{\varrho}_j + \frac{c_\ell + c_r}{c_j}(\hat{\mu}_\ell + \hat{\mu}_r) - \mu_j\right)^2\right] \\
&= \frac{c_j - c_\ell - c_r}{c_j}\mathbb{E}\left[(\hat{\varrho}_j - \mu_j)^2\right] + \frac{c_\ell + c_r}{c_j}\left(\mathbb{E}\left[(\hat{\mu}_\ell - \mu_\ell)^2\right] + \mathbb{E}\left[(\hat{\mu}_r - \mu_r)^2\right]\right) \\
&= \frac{c_j - c_\ell - c_r}{c_j}\frac{\varsigma_j^2}{M_j - M_\ell - M_r} + \frac{c_\ell + c_r}{c_j}\left(\sigma_\ell^2 + \sigma_r^2\right) \qquad (11)
\end{aligned}$$

[1] Strictly speaking, this can be violated for any estimates passed *down* through the tree when using SMC.

where M_j is the number of times the node has been traversed (as opposed to N_j which is the number of times the node has been run), ς_j^2 is the variance in the estimate produced a single run (e.g SMC sweep) at node j, and we have omitted an implicit dependency of the expectations on M_ℓ, M_r and M_j. The interpretation of σ_j is as a standard error in the mean estimate, i.e. that it is the root MSE for the estimate $\hat{\mu}_j$, given the number of samples that have been taken and the node weights. We can thus use an empirical estimate $\hat{\sigma}_j$ as a measure of uncertainty in the mean estimate at a node. For this we only need to be able to calculate

$$\hat{\varsigma}_j^2 = \left(\frac{M_j - M_\ell - M_r}{M_j - M_\ell - M_r - 1}\right)\left(\left(\frac{1}{M_j - M_\ell - M_r}\sum_{m=1}^{M_j - M_\ell - M_r} \hat{\varrho}_{j,m}^2\right) - \hat{\varrho}_j^2\right) \tag{12}$$

where $\varrho_{j,m}$ is the mean estimate for the m^{th} run (e.g. SMC sweep) at node j, $\hat{\varrho}_j = \frac{1}{M_j - M_\ell - M_r}\sum_{m=1}^{M_j - M_\ell - M_r} \hat{\varrho}_{j,m}$, and $\left(\frac{M_j - M_\ell - M_r}{M_j - M_\ell - M_r - 1}\right)$ is Bessel's correction. We now define our empirical estimates for standard error by taking $\hat{\sigma}_j^2 = \hat{\varsigma}_j^2$ for the leaf nodes, and using (11) to calculate estimates for all others through propagation.

For each node, we now have an estimate for the marginal probability $\hat{\omega}_j$ and an uncertainty estimate for this estimate, $\hat{\sigma}_j$. We use these as the basis of an upper confidence bounding scheme for the traversal. Namely, we calculate a utility for the left child

$$u_\ell = \frac{1}{M_\ell}\left(\left(\frac{\hat{\omega}_\ell + u\hat{\sigma}_\ell}{\hat{\omega}_j}\right)^a + \frac{b\log(M_\ell + M_r + N_j)}{\sqrt{M_\ell}}\right) \tag{13}$$

and similarly for the right child. Here $a \leq 1$ is an annealing parameter that is set to encourage more exploration, noting that the empirical uncertainty estimates are often substantial underestimates. We set a to increase with the number of IT iterations K. The second term in the above is an upper confidence term as per [6], where $b > 0$ is a parameter that encourages more exploration when increased. We further define a utility for the parent node as

$$u_j = \frac{1}{N_j}\left(\frac{b'\log(M_\ell + M_r + N_j)}{\sqrt{N_j}}\right). \tag{14}$$

The rationale for this is that if $\hat{\omega}_j$ is substantially larger than $\hat{\omega}_\ell$ and $\hat{\omega}_r$, then it will be unclear whether to traverse left or right as both estimates are very poor, such that it is perhaps best to instead run the current node and propagate these samples down through the tree. Note that b' plays the same role as b in (13) but can have a different numeric value.

Our three utilities, u_ℓ, u_r, and u_j, reflect how relatively often we should choose each action relative to how often we already have, hence, for example, the M_ℓ term at the front of (13). The optimal action is thus the one with the highest utility, which is deterministically chosen to complete the definition of our core traversal strategy. We note, however, that we employ the important heuristic of turning the bandit strategy off at every other iteration to allow acceleration of the tree growth when certain areas have substantially better areas than others. To ensure the effect is not immediately undermined, the M_j and N_j terms in (13) and (14) are based only on the number of runs / traversals when the bandit strategy is switched on.

C.1.2 Known f

When f is known, we desire to exploit this information by directly trying to variance of the estimator $\sigma^2 = \mathbb{E}\left[(\hat{\mu} - \mu)^2\right] = \mathbb{E}\left[\hat{\mu}^2\right] - \left(\mathbb{E}\left[\hat{\mu}\right]\right)^2$. As in the case where f was unknown, we can exploit the fact that variances for different nodes are additive as per (11) to propagate empirical estimates for the mean squared error at each node $\hat{\sigma}_j^2$ up the tree. We can then define a loss function for a particular choice in the relative frequency of choosing two children as

$$L_M(M_\ell, M_r) = \mathbb{E}\left[(\hat{\mu}_\ell - \mu_\ell)^2 \middle| M_\ell, M_r\right] + \mathbb{E}\left[(\hat{\mu}_r - \mu_r)^2 \middle| M_\ell, M_r\right] \tag{15}$$

$$= \mathbb{E}\left[\left(\frac{1}{M_\ell}\sum_{m=1}^{M_\ell} \hat{\mu}_{\ell,m} - \mu_\ell\right)^2 \middle| M_\ell, M_r\right] + \mathbb{E}\left[\left(\frac{1}{M_r}\sum_{m=1}^{M_r} \hat{\mu}_{r,m} - \mu_r\right)^2 \middle| M_\ell, M_r\right] \tag{16}$$

$$= \frac{1}{M_\ell}\mathbb{E}\left[(\hat{\mu}_{\ell,1} - \mu_\ell)^2 \middle| M_\ell, M_r\right] + \frac{1}{M_r}\mathbb{E}\left[(\hat{\mu}_{r,1} - \mu_r)^2 \middle| M_\ell, M_r\right] = \frac{s_\ell^2}{M_\ell} + \frac{s_r^2}{M_r} \tag{17}$$

where s_j^2 represents a "per-sample" MSE (i.e. standard error squared). Given a budget $M = M_\ell + M_r$ it is easily shown (see e.g. [6]) that $L_M(M_\ell, M_r)$ is minimized when $\frac{M_\ell}{M_r} = \frac{s_\ell}{s_r}$, for which we have $M_\ell = \frac{s_\ell M}{s_\ell + s_r}$, $M_r = \frac{s_r M}{s_\ell + s_r}$, and $L_M(M_\ell, M_r) = \frac{(s_\ell + s_r)^2}{M}$. The optimal static strategy is thus to sample the left child with probability. $P_\ell^* = \frac{s_\ell}{s_\ell + s_r}$. Note that s_ℓ and s_r are fixed but unknown values. Strictly speaking, the bias of the IT algorithm means that they are each functions of both M_ℓ and M_r, but in the limit of large sample size, they tend to the true per-sample MSE of running the estimator in isolation, i.e. each $s_j \to \sqrt{\mathbb{E}\left[(\hat{\mu}_{j,1} - \mu_j)^2\right]}$.

In similar way as done in [6], we can thus evaluate a regret for any strategy of choosing children as the loss a strategy occurs minus the loss incurred by an oracle that knows s_ℓ and s_r upfront and uses the optimal static strategy. Namely, we can say that the regret $R_M(\mathcal{S})$ for a strategy $\mathcal{S}$ is

$$R_M(\mathcal{S}) = L_M(\mathcal{S}) - \frac{(s_\ell + s_r)^2}{M} \tag{18}$$

noting that as the loss is defined as a MSE, this is a deterministic value for a given problem and strategy. As noted by [6], any strategy is asymptotically equivalent to the optimal strategy if its regret is asymptotically negligible compared to $1/M$.

Unlike in the case were f is unknown, we do not also have an error estimate for our target evaluations (because our target is itself the error estimate we used before) and so cannot include an explicit uncertainty estimate. However, other than omitting this uncertainty estimate term, we take the same upper confidence bound (UCB) approach as the unknown f case, replacing, for example, $\left(\frac{\hat{\omega}_\ell + u\hat{\sigma}_\ell}{\hat{\omega}_j}\right)^a$ with $\left(\frac{\hat{s}_\ell}{\hat{s}_j}\right)^a$ in (13). We note that [6] provide finite-time regret bounds when the estimator output is sub-Gaussian (i.e. distribution whose tails are lighter than a Gaussian) for an approach closely related to ours, but which requires a total sampling budget to be set. Those these bounds do not directly apply, we still expect them to be indicative up to constant factors.

C.2 Refinement

Though not necessary for convergence, choosing good splits can be highly beneficial to the performance of the IT algorithm, while some splits can hinder performance. For example, if we make a split with roughly even posterior mass in each child branch (presuming we are in the unknown f case, as we will in the rest of the section), it will be necessary to traverse both branches and little is gained from the split. At a high-level, a good partitioning structure is one in which the posterior mass is concentrated in a small number of regions. In essence, we gain most from being able to "eliminate regions" from our consideration, reducing the proportion of the target space that needs to be actively considered. When we propose to split a node, we thus want to find the split that best concentrates the posterior mass of that mode.

Conveniently, we can use the samples already generated at the node to try and predict what will be a good split. Namely, we construct a split criterion based on an entropy metric and empirically estimate this metric using the samples already at the node. Hypothetical splits can be quickly tested in this way – there is no need to run any new inference – and so we propose a relatively large number of splits uniformly at random (100 taken as default), estimate our entropy metric for each of these suggested splits, and then choose the one with the best estimate, namely that with the minimum entropy.

To define this entropy metric more precisely, recall that the entropy of continuous uniform distribution $U(s_1, s_2)$ is

$$\textsc{Entropy}(U(s_1, s_2)) = -\int_{s_1}^{s_2} p(x) \log p(x) \, \mathrm{d}x = \ln(s_2 - s_1). \tag{19}$$

Assume that we propose a split at a point $s \in (s_1, s_2)$, and that we will later go to the left of this split with a probability P_ℓ and to the right with a probability $P_r = 1 - P_\ell$. This splitting and the traversal strategy give rise to a proposal of a mixture of two uniform distributions that has the following density

$$p_s(x) = \begin{cases} d_\ell & \text{if } x < s, \text{ where } d_\ell = \frac{P_\ell}{s - s_1} \\ d_r & \text{otherwise, where } d_r = \frac{1 - P_\ell}{s_2 - s} \end{cases} \tag{20}$$

The entropy of this proposal is:

$$\textsc{Entropy}(p_a) = -\int_{s_1}^{s} d_\ell \log d_\ell \, \mathrm{d}x - \int_{s}^{s_2} d_r \log d_r \, \mathrm{d}x = P_\ell \log \frac{1}{d_\ell} + P_r \log \frac{1}{d_r}. \tag{21}$$

We can now use our empirical estimates $\hat{\omega}_\ell = \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}(x_i \in B_\ell)w_i$ and similarly $\hat{\omega}_r = \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}(x_i \notin B_\ell)w_i$ to define our entropy metric as

$$\textsc{Entropy}(p_s) = \hat{P}_\ell \log \frac{s - s_1}{\hat{P}_\ell} + \hat{P}_r \log \frac{s_2 - s}{\hat{P}_r} \quad \text{where} \quad \hat{P}_\ell = \frac{\hat{\omega}_\ell}{\hat{\omega}_\ell + \hat{\omega}_r}, \quad \hat{P}_r = \frac{\hat{\omega}_r}{\hat{\omega}_\ell + \hat{\omega}_r}.$$

We then choose the split $s^* = \arg\min_s \textsc{Entropy}(p_s)$ where the minimization is over our randomly sampled candidate splits.

After choosing the best split among all candidates and separating the space in to B_ℓ and B_r, we run inference restricted to B_ℓ and B_r separately. Then we compare the empirical estimates of the marginal likelihood for each child using a t-test, which shows how likely the results are samples from two different distributions. If the p-value is small, it suggests the split is meaningful. In that case, we accept the split, creating two new child nodes and converting the current leaf node to a discriminant note. Otherwise, we discard the split and combine the samples, adding them to the estimate of the current node. When the node is revisited, new splits are suggested and the process continues in the same way.

D Experiment Details

In this section, we provide further details for the experiments laid out in Section D. Our first experiment is a non-linear chaotic system with four latent global parameters. Our second experiment is a Gaussian mixture model, which has the symmetry and multiple modes. Our third, and final, experiment is a network model where we approximate the expectation of a known target function f. In all cases, the provided ground truth estimates are estimated using a multiple SMC runs with a large number of particles separately on a number of carefully chosen manual partitions, such that one of these targets a small area containing the vast majority of the target posterior mass. Estimates from the partitions are combined to a single overall partition in a similar manner to stratified sampling (and ITs). Because the area with significant posterior mass is a small proportion of the overall input space for all the problems, this method allows a very low variance estimate to be found for this partition, while the contribution of the other partitions to the overall estimate is negligible, giving an low variance overall estimate.

D.1 Chaos Model

We first consider a problem based on tracking a dynamical system with chaotic dynamics that was first introduced by [18]. Assume that we observe a noisy signal $y_t \in \mathbb{R}^K$ for $t = 1, 2, \ldots$ in some K dimensional observation space with some underlying latent space $x_t \in \mathbb{R}^D$ which evolves according to complicated non-linear dynamics. Given observations up to some time T, we will use an extended Kalman filter model as defined by

$$\begin{aligned} x_0 &\sim \mathcal{N}(\mu_0, \sigma_0 I) \\ f_t(x_t|x_{t-1}) &= A(x_{t-1}, \theta) + \delta_{t-1}, \quad \delta_{t-1} \sim \mathcal{N}(0, \sigma_q I) \\ g_t(y_t|x_t) &= Cx_t + \varepsilon_t, \quad \varepsilon_t \sim \mathcal{N}(0, \sigma_y I). \end{aligned}$$

Here I is the identity matrix, C is a known $K \times D$ matrix, μ_1 is the expected starting position, and σ_0, σ_q and σ_y are all scalars which are assumed to be known. The transition function $A(\cdot, \theta)$ dictates the underlying dynamics with parameters θ. We will assume that the form of A is known but not the parameters. Namely, we consider the example where the dynamics correspond to the Pickover attractor defined as

$$\begin{aligned} x_{t,1} &= \sin(\beta x_{t-1,2}) - \cos(\alpha x_{t-1,1})\, x_{t-1,3} \\ x_{t,2} &= \sin(\gamma x_{t-1,1})\, x_{t-1,3} - \cos(\eta x_{t-1,2}) \\ x_{t,3} &= \sin(x_{t-1,1}) \end{aligned}$$

where $\theta = (\beta, \alpha, \gamma, \eta)$ and $D = 3$. We finish the model by defining the following separable uniform prior on the parameters

$$\beta \sim \textsc{Uniform}(-3, 3), \quad \alpha \sim \textsc{Uniform}(0, 3), \quad \gamma \sim \textsc{Uniform}(-3, 3), \quad \eta \sim \textsc{Uniform}(0, 3).$$

We desire to conduct inference over both the parameters and the latent variables.

To make comparisons, we generated a single synthetic dataset using $K = 20$, $T = 500$, $\mu_0 = [0, 0, 0]$, $\sigma_0 = 1$, $\sigma_q = 0.01$, $\sigma_y = 0.2$, a fixed matrix C where each column was randomly drawn from a symmetric Dirichlet distribution with parameter 0.1, and ground truth transition parameters $\beta = -2.3$, $\alpha = 2.5$, $\gamma = -1.5$ and $\eta = 1.25$. We then ran ITs using an SMC base inference algorithm with

bootstrap proposals where we restricted the variables controlled by the tree to x_0 and θ (giving a 7 dimensional problem), such that the full proposal is used for the other variables at all times.

Figure 2(a) compares the estimation of marginal likelihood of the IT with that of SMC under the same total budget (4 million samples). Note that the time taken in training the IT was dominated by that running SMC, such that the x-axis is effectively equivalent to run time.

D.2 Gaussian Mixture Model

We next consider a Gaussian mixture model (GMM), i.e. we consider a clustering problem where we use a model that assumes that each data point $\mathbf{y}_n$ is independently sampled from one of K Gaussian distributions, each with mean $\boldsymbol{\mu}_k$ and precision matrix $\boldsymbol{\Lambda}_k$. A latent variable $z_n \in \{1, \ldots, K\}$ is used to indicate which cluster the datapoint belongs to, i.e. which Gaussian it was generated from. We presume that $\boldsymbol{\Lambda}_k$ is known, but place a prior on $\boldsymbol{\mu}_k$ and the cluster assignments z_n as follows

$$\begin{aligned} \boldsymbol{\mu}_k &\sim \text{Normal}(\mathbf{0}, (\beta\boldsymbol{\Lambda}_k)^{-1}) \\ z_n \big| \, \boldsymbol{\pi} &\sim \text{Categorical}(\{1/K, \ldots, 1/K\}) \\ \mathbf{y}_n \big| \, z_n = k, \boldsymbol{\mu}_k &\sim \text{Normal}(\boldsymbol{\mu}_k, \boldsymbol{\Lambda}_k^{-1}) \end{aligned}$$

completing our model definition. Note that is possible to analytically marginalize over the z_n, such that, for the purposes of leaning $\boldsymbol{\mu}_k$, we can analytically calculate $p\left(\mathbf{y}_{1:N} | \boldsymbol{\mu}_{1:K}\right)$ directly.

For this model we compared using IT with importance sampling as the base inference algorithm (noting that SMC and importance sampling are identical for this problem) to just using importance sampling directly. We generated synthetic data using the model with $K = 3$ clusters and $N = 300$ total datapoints. The inference problem is to calculate $p\left(\boldsymbol{\mu}_{1:K} | \mathbf{y}_{1:N}\right)$ and we used ITs to control the sampling of all 6 variables. Note that we used a different $\boldsymbol{\Lambda}_k$ for each cluster, breaking the symmetry that is common in GMM models such that the model is no longer invariant to permutation of the cluster indices.

Results are given in Figure 2(b). Even though ITs performed much better than importance sampling given equivalent computation resources, it sometimes substantially underestimated the marginal likelihood. The GMM posterior has multiple modes of varying magnitude, resulting from the possible permutations of cluster indices. It is clear that the tree does not always "find" all such modes and in particular, sometimes "misses" the largest mode, i.e. generating few, or even no, samples close to its peak. We believe the key underlying reason for this is that the uncertainty estimates produced during the IT algorithm are insufficient to adequately convey the possibility of a missing mode. We are currently, therefore, trying to alleviate this problem by developing more advanced uncertainty estimate schemes using extreme value theory.

D.3 Network Model

Finally, we consider a network model with weighted edges where we wish to estimate if the shortest path between two points exceeds a threshold. One possible application of such models would be in modeling a traffic network, where the edges are streets connecting two points and the weights correspond to the commuting times on different edges which are stochastic due to traffic levels and correlated because of the proximity of different streets to one another. We thus assume that there are noisy, correlated, observations for the edges weights, requiring inference, while our threshold function means we are in a "known f" scenario, namely we are estimating a form of tail integral.

The model is formally defined as

$$x_{1:T} \sim \mathcal{N}(x_{1:T}; \mu, \Sigma) \tag{22}$$

$$y_t | x_t \sim \text{STUDENT-T}\left(\frac{y_t - x_t}{\sigma}; \nu\right) \quad \forall t \in \{1, \ldots, T\} \tag{23}$$

where x_t represents the unknown weights of edges, y_t are noisy observations of those weights, and μ, Σ, ν, and σ are known fixed parameters. Synthetic data was generated by setting $T = 10$, $\mu = [3, \ldots, 3]$, $\Sigma = I$ (i.e. the identity matrix), $\sigma = 0.1$, and $\nu = 5$. We take the threshold as 3.8 and look to estimate the probability that the shortest path exceeds this threshold, which in our traffic analogy would correspond to not being able to reach a destination on time. We note that this corresponds to a tail integral problem. Figure 2(c) shows that IT outperform both SMC with the same number of samples and SMC with 1000 times more samples.