Journal of Computational Physics 496 (2024) 112577

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

Efficient multifidelity likelihood-free Bayesian inference with adaptive computational resource allocation

Thomas P. Prescott [a,b], David J. Warne [c], Ruth E. Baker [d,*]

[a] Alan Turing Institute, London NW1 2DB, United Kingdom
[b] Datasparq, 29 Clerkenwell Road, London EC1M 5RN, United Kingdom
[c] School of Mathematical Sciences, Queensland University of Technology, Brisbane, QLD 4000, Australia
[d] Mathematical Institute, University of Oxford, Oxford OX2 6GG, United Kingdom

ARTICLE INFO

Keywords:
Likelihood-free Bayesian inference
Multifidelity approaches

ABSTRACT

Likelihood-free Bayesian inference algorithms are popular methods for inferring the parameters of complex stochastic models with intractable likelihoods. These algorithms characteristically rely heavily on repeated model simulations. However, whenever the computational cost of simulation is even moderately expensive, the significant burden incurred by likelihood-free algorithms leaves them infeasible for many practical applications. The multifidelity approach has been introduced in the context of approximate Bayesian computation to reduce the simulation burden of likelihood-free inference without loss of accuracy, by using the information provided by simulating computationally cheap, approximate models in place of the model of interest. In this work we demonstrate that multifidelity techniques can be applied in the general likelihood-free Bayesian inference setting. Analytical results on the optimal allocation of computational resources to simulations at different levels of fidelity are derived, and subsequently implemented practically. We provide an adaptive multifidelity likelihood-free inference algorithm that learns the relationships between models at different fidelities and adapts resource allocation accordingly, and demonstrate that this algorithm produces posterior estimates with near-optimal efficiency.

1. Introduction

Across domains in engineering and science, parametrised mathematical models are often too complex to analyse directly. Instead, many *outer-loop applications* [1], such as model calibration, optimization, and uncertainty quantification, rely on repeated simulation to understand the relationship between model parameters and behaviour. In time-sensitive and cost-aware applications, the typical computational burden of such simulation-based methods makes them impractical. Multifidelity methods, reviewed by Peherstorfer et al. [1,2] and Ng and Willcox [3], are a family of approaches that exploit information gathered from simulations, not only of a single model of interest, but also of additional approximate or surrogate models. In this article, the term *model* refers to the underlying mathematical abstraction of a system in combination with the computer code used to implement simulations. Thus, 'model approximation' may refer to mathematical simplifications and/or approximations in numerical methods. The fundamental challenge when implementing multifidelity techniques is the allocation of computational resources between different models, for the purposes of balancing a characteristic trade-off between maintaining accuracy and saving computational burden.

* Corresponding author.
E-mail address: ruth.baker@maths.ox.ac.uk (R.E. Baker).

https://doi.org/10.1016/j.jcp.2023.112577
Received 16 February 2023; Received in revised form 13 October 2023; Accepted 16 October 2023

Available online 23 October 2023

In this work, we consider a specific outer-loop application that arises in Bayesian statistics, the goal of which is to calibrate a parametrised model against observed data. Bayesian inference uses the likelihood of the observed data to update a prior distribution on the model parameters into a posterior distribution, according to Bayes's rule. In the situation where the likelihood of the data cannot be calculated, we rely on so-called likelihood-free methods that provide estimates of the likelihood by comparing model simulations to data. For example, approximate Bayesian computation (ABC) is a widely-known likelihood-free inference technique [4,5] where the likelihood is typically estimated as a binary value that records whether or not the distance between a simulation and the observed data falls within a given threshold. Other likelihood-free methods are also available, such as pseudo-marginal methods and Bayesian synthetic likelihoods (BSL) [6]. In this work, we develop a generalised likelihood-free framework for which ABC, pseudo-marginal and BSL can be expressed as specific cases, as described in Section 2.1.

The significant cost of likelihood-free inference has motivated several successful proposals for improving the efficiency of likelihood-free samplers, such as ABC-MCMC [7] and ABC-SMC [8–10]. These approaches aim to efficiently explore parameter space by avoiding the proposal of low-likelihood parameters, reducing the required number of expensive simulations required and improving the ABC acceptance rate. However, an 'orthogonal' technique for improving the efficiency of likelihood-free inference is to instead ensure that each simulation-based likelihood estimate is, on average, less computationally expensive to generate.

In previous work, Prescott and Baker [11,12] investigated multifidelity approaches to likelihood-free Bayesian inference [13], with a specific focus on ABC [4,5]. Suppose that there exists a low-fidelity approximation to the parametrised model of interest, and that the approximation is relatively cheap to simulate. Monte Carlo estimates of the posterior distribution, with respect to the likelihood of the original high-fidelity model, can be constructed using the simulation outputs of the low-fidelity approximation. Prescott and Baker [11] showed that using the low-fidelity approximation introduces no further bias, so long as, for any parameter proposal, there is a positive probability of simulating the high-fidelity model to check and potentially correct a low-fidelity likelihood estimate. The key to the success of the multifidelity ABC (MF-ABC) approach is to choose this positive probability to be suitably small, thereby simulating the original model as little as possible, while ensuring it is large enough that the variance of the resulting Monte Carlo estimate is suitably small. The result of the multifidelity approach is to reduce the expected cost of estimating the likelihood for each parameter proposal in any Monte Carlo sampling algorithm. In subsequent work, Prescott and Baker [12] showed that this approach integrates with sequential Monte Carlo (SMC) sampling for efficient parameter space exploration [9,10,14], and Warne et al. [15] demonstrated its applicability to multilevel Monte Carlo Guha and Tan [16], Jasra et al. [17], Warne et al. [18]. Thus, the synergistic effect of combining multifidelity with other Monte Carlo schemes to improve the efficiency of ABC has been demonstrated.

Multifidelity ABC can be compared with previous techniques for exploiting model approximation in ABC, such as Preconditioning ABC [19], Lazy ABC (LZ-ABC) [20], and Delayed Acceptance ABC (DA-ABC) [21,22]. The preconditioning approach seeks to explore parameter space more efficiently, by proposing parameters for high-fidelity simulation with greater low-fidelity posterior mass. In contrast, each of MF-ABC, LZ-ABC, and DA-ABC seeks to make each parameter proposal quicker to evaluate, on average, by using the output of the low-fidelity simulation to directly decide whether to simulate the high-fidelity model. In both LZ-ABC and DA-ABC, a parameter proposal is either (a) rejected early, based on the simulated output of the low-fidelity model, or (b) sent to a high-fidelity simulation, to make a final decision on ABC acceptance or rejection. The distinctive aspect of MF-ABC is that step (a) is different; it is not necessary to reject early to avoid high-fidelity simulation. Instead the low-fidelity simulation can be used to make the accept/reject decision directly. In both DA-ABC and MF-ABC, the decision between (a) or (b) is based solely on whether the low-fidelity simulation would be accepted or rejected. In contrast, LZ-ABC allows for a much more generic decision of whether to simulate the high-fidelity model, requiring an extensive exploration of practical tuning methods.

More generally, there are multifidelity methods that exploit tractable surrogate models and apply subsequent adaptations or transformations to correct for bias in this surrogate. For example, Yan and Zhou [23,24] adaptively tune surrogate models based on polynomial chaos or deep learning methods, and Bon et al. [25] use a population of surrogates and moment-matching transformations in a similar sense to Warne et al. [19]. While these approaches are of interest, we primarily focus on multifidelity schemes that are strictly simulation-based. However, we note that surrogate likelihood approaches can also be expressed within our theoretical framework.

In this paper, we show that the multifidelity approach can be applied to any simulation-based likelihood-free inference methodology, including but not limited to ABC. We achieve this by developing a generalised framework for likelihood-free inference, and deriving a multifidelity method to operate in this framework. A successful multifidelity likelihood-free inference algorithm requires us to determine how many simulations of the high-fidelity model to perform, based on the parameter value and the simulated output of the low-fidelity model. We provide theoretical results and practical, automated tuning methods to allocate computational resources between two models, designed to optimise the performance of multifidelity likelihood-free importance sampling.

1.1. Outline

In Section 2 we introduce a generalised framework for likelihood-free Bayesian inference for which standard approaches are shown to be a special case. In Section 3 a general multifidelity likelihood-free importance sampler is constructed based on the MF-ABC approach of Prescott and Baker [11]. This section also explores how to practically allocate computation between model fidelities, by adaptively evolving the allocation in response to learned relationships between simulations at each fidelity across parameter space. Analysis is presented in Section 4, including the proofs of the main results set out in Section 3.3, in which we determine the optimal allocation of computational resources between the two models to achieve the best possible performance of multifidelity inference. We illustrate adaptive multifidelity inference by applying the algorithm to a fundamental biochemical

network motif in Section 5. We show that, using a low-fidelity Michaelis–Menten approximation together with the exact model (both simulated using the exact algorithm of [26]) our adaptive implementation of multifidelity likelihood-free inference can achieve a quantifiable speed-up in constructing posterior estimates to a specified variance and with no additional bias. Code for this example, developed in Julia 1.6.2 [27], is available at `github.com/tpprescott/mf-lf`. Finally, in Section 6 we discuss how greater improvements may be achieved for more challenging inference tasks.

2. Likelihood-free inference

We consider a stochastic model of the data generating process, defined by a distribution with parametrised probability density function, $f(\cdot \mid \theta)$, where the parameter vector θ takes values in a parameter space Θ. For any $\theta \in \Theta$, the model induces a probability density, denoted $f(y \mid \theta)$, on observable outputs, with y taking values in an output space $\mathcal{Y}$. We note that the model is usually implemented in computer code to allow simulation, through which outputs $y \in \mathcal{Y}$ can be generated. We write $y \sim f(\cdot \mid \theta)$ to denote simulation of the model f given parameter values θ. Taking the experimentally observed data $y_0 \in \mathcal{Y}$, we define the likelihood function to be a function of θ using the density, $\mathcal{L}(\theta) = f(y_0 \mid \theta)$, of the observed data under this model.

Bayesian inference updates prior knowledge of the parameter values, $\theta \in \Theta$, which we encode in a prior distribution with density $\pi(\theta)$. The information provided by the experimental data, encoded in the likelihood function, $\mathcal{L}(\theta)$, is combined with the prior using Bayes' rule to form a posterior distribution, with density

$$\pi(\theta \mid y_0) = \frac{\mathcal{L}(\theta)\pi(\theta)}{Z},$$

where $Z = \int \mathcal{L}(\theta)\pi(\theta)\,\mathrm{d}\theta$ normalises $\pi(\cdot \mid y_0)$ to be a probability distribution on Θ. For a given, arbitrary, integrable function $G : \Theta \to \mathbb{R}$, we take the goal of the inference task as the production of a Monte Carlo estimate of the posterior expectation,

$$\mathbf{E}(G \mid y_0) = \int G(\theta)\pi(\theta \mid y_0)\,\mathrm{d}\theta,$$

conditioned on the observed data.

2.1. Approximating the likelihood with simulation

In most practical settings, models tend to be sufficiently complicated that calculating $\mathcal{L}(\theta) = f(y_0 \mid \theta)$ for $\theta \in \Theta$ is intractable. In this case, we exploit the ability to produce independent simulations from the model, $\mathbf{y} = (y_1, \dots, y_K)$ with $y_k \sim f(\cdot \mid \theta)$. In the following, we will slightly abuse notation by using the shorthand $\mathbf{y} \sim f(\cdot \mid \theta)$ to represent K independent, identically distributed draws from the parametrised distribution $f(\cdot \mid \theta)$.

Given the observed data, y_0, we can define a real-valued function, referred to as a *likelihood-free weighting*, $\omega : (\theta, \mathbf{y}) \mapsto \mathbb{R}$, which varies over the joint space of parameter values and simulation outputs. Here, ω is a function of a parameter value, θ, and a vector, $\mathbf{y}$, of stochastic simulations. For a fixed θ, we can take conditional expectations of $\omega(\theta, \mathbf{y})$ over the probability density of simulations, $f(\mathbf{y} \mid \theta)$, to define an *approximate likelihood function*,

$$L_\omega(\theta) = \mathbf{E}(\omega \mid \theta) = \int \omega(\theta, \mathbf{y}) f(\mathbf{y} \mid \theta)\,\mathrm{d}\mathbf{y}, \tag{1}$$

where $\omega(\theta, \mathbf{y})$ is chosen such that $L_\omega(\theta)$ is an appropriate approximation to the modelled likelihood function, $\mathcal{L}(\theta)$. The determination of what is appropriate as an approximation will depend on the implementation or application. For example, weights corresponding to BSL will not be appropriate if the distribution of the summary statistics are highly non-Gaussian. Similarly, within an ABC setting, it is not appropriate to choose a very large discrepancy threshold, as this will lead to $L_\omega(\theta) \approx 1$ for any θ due to very few proposals being rejected. For standard likelihood-free methods, such as ABC, BSL and pseudo-marginal methods, the likelihood-free weighting, $\omega(\theta, \mathbf{y})$, is a random variable (since it is a function of K stochastic simulations) and a Monte Carlo estimate of the approximate likelihood function, $L_\omega(\theta)$. The explicit form of this Monte Carlo estimate is implementation specific (see Appendix A). While we focus here on the Monte Carlo setting, it is worth highlighting that $\omega(\theta, \mathbf{y})$ may be treated as a deterministic function of θ (thus independent of any stochastic simulations) in order to implement a likelihood function, surrogate likelihood function, or some alternative loss function. For example, if $\omega(\theta, \mathbf{y}) = f(y_0 \mid \theta)$ then Equation (1) reduces to $L_\omega(\theta) = \mathcal{L}(\theta)$.

The approximate likelihood function is used to define the *likelihood-free approximation to the posterior*,

$$\pi_\omega(\theta \mid y_0) = \frac{L_\omega(\theta)\pi(\theta)}{Z_\omega},$$

with the normalisation constant $Z_\omega = \int_\Theta L_\omega(\theta)\pi(\theta)\,\mathrm{d}\theta$. The likelihood-free approximation to the posterior, $\pi_\omega(\theta \mid y_0)$, subsequently induces a potentially biased approximation of $\mathbf{E}(G \mid y_0)$, given by

$$\mathbf{E}_{\pi_\omega}(G \mid y_0) = \int G(\theta)\pi_\omega(\theta \mid y_0)\,\mathrm{d}\theta.$$

In this situation, the success of likelihood-free inference depends on ensuring that the likelihood-free weighting, $\omega(\theta, \mathbf{y})$ is chosen such that the squared difference $(\mathbf{E}(G \mid y_0) - \mathbf{E}_{\pi_\omega}(G \mid y_0))^2$ between the posterior expectation, $\mathbf{E}(G \mid y_0)$, and its likelihood-free approximation, $\mathbf{E}_{\pi_\omega}(G \mid y_0)$, is as small as possible. Most importantly, the accuracy of the likelihood-free approximation is entirely encoded

in the weighting function, and in most, but not all, cases this will be biased. For example, standard likelihood-free methods such as ABC, BSL and pseudo-marginal methods can be implemented using different choices for $\omega(\theta, \mathbf{y})$ (see Appendix A), however, only ABC and BSL introduce bias.

2.2. Likelihood-free importance sampling

A direct approach to estimating the likelihood-free approximate posterior expectation, $\mathbf{E}_{\pi_\omega}(G \mid y_0)$, is to use importance sampling. We assume that parameter proposals $\theta_i \sim q(\cdot)$, for $i = 1, \ldots, N$, can be sampled from a given importance distribution, the support of which must include the prior support, that is, $q(\theta) \neq 0$ if $\pi(\theta) > 0$. In practice, we need only know the importance density, $q(\theta)$, up to a multiplicative constant. We also assume that we have access to the prior probability density, $\pi(\theta)$.

The *likelihood-free importance sampling* algorithm is described in Algorithm 1. This algorithm requires the specification of an importance distribution, q, and a likelihood-free weighting, $\omega(\theta, \mathbf{y})$, with conditional expectation, $L_\omega(\theta) = \mathbf{E}(\omega \mid \theta)$. The output of Algorithm 1, $\hat{G}$, is an estimate of the likelihood-free approximate posterior expectation, $\mathbf{E}_{\pi_\omega}(G \mid y_0)$. We show (Section 4, Theorem 1), the standard result that $\hat{G}$ is a *consistent* estimate of $\mathbf{E}_{\pi_\omega}(G \mid y_0)$, and quantify the dominant behaviour of the mean-squared error (MSE) in the limit of large sample sizes, $N \to \infty$.

Algorithm 1 Likelihood-free importance sampling.

Require: Prior, π; importance distribution, q; likelihood-free weighting, ω; model $f(\cdot \mid \theta)$; target function, G.

for $i \in [1, 2, \ldots, N]$ **do**

Sample $\theta_i \sim q(\cdot)$;

Simulate $\mathbf{y}_i \sim f(\cdot \mid \theta_i)$;

Calculate weight $w_i = w(\theta_i, \mathbf{y}_i) = \dfrac{\pi(\theta_i)}{q(\theta_i)}\omega(\theta_i, \mathbf{y}_i)$;

end for

Estimate expectation using weighted sum, $\hat{G} = \dfrac{\sum_{i=1}^N w_i G(\theta_i)}{\sum_{j=1}^N w_j}$.

In obtaining the consistency result, we also determine the leading-order behaviour of the MSE of the output of Algorithm 1 in terms of sample size,

$$\mathbf{E}\left((\hat{G} - \mathbf{E}_{\pi_\omega}(G \mid y_0))^2\right) = \left[\frac{\mathbf{E}\left(W^2(G(\theta) - \mathbf{E}_{\pi_\omega}(G \mid y_0))^2\right)}{\mathbf{E}(W)^2}\right]\frac{1}{N} + O\left(\frac{1}{N^2}\right), \tag{2}$$

where W is the random variable for the value of the importance weight, w, as defined in Algorithm 1. For later formulations it is more useful to denote c_i as the random computational cost of the K stochastic simulations, $\mathbf{y}_i \sim f(\cdot|\theta_i)$, then consider a fixed computational budget, C_{tot}, rather than a fixed sample size. That is, we take N as the largest index i for which $\sum_{j=1}^i c_j \leq C_{\text{tot}}$. We can also quantify the performance of this algorithm in terms of how the MSE decreases with increasing the overall computational budget. To leading-order this is given by (Section 4, Corollary 2)

$$\mathbf{E}\left((\hat{G} - \mathbf{E}_{\pi_\omega}(G \mid y_0))^2\right) = \left[\frac{\mathbf{E}(C)\mathbf{E}\left(W^2(G(\theta) - \mathbf{E}_{\pi_\omega}(G \mid y_0))^2\right)}{\mathbf{E}(W)^2}\right]\frac{1}{C_{\text{tot}}} + O\left(\frac{1}{C_{\text{tot}}^2}\right). \tag{3}$$

We can use the leading-order coefficient of $1/C_{\text{tot}}$ in Equation (3) to quantify the performance of likelihood-free importance sampling. Importantly, this expression explicitly depends on the expected computational cost, C, of each iteration of Algorithm 1 and the variance of the weighted errors, $w(G(\theta) - \mathbf{E}_{\pi_\omega}(G \mid y_0))$. In the importance sampling context, the optimal importance distribution q should seek to minimise this coefficient. This is achieved by minimisation of the numerator, $\mathbf{E}(C)\mathbf{E}\left(W^2(G(\theta) - \mathbf{E}_{\pi_\omega}(G \mid y_0))^2\right)$, that is, by trading off a preference for parameter values with lower computational burden against ensuring small variability in the weighted errors. However, in addition to tuning the proposal mechanism, q, Equation (3) also provides insight into how approximations might be used to directly reduce the leading-order coefficient in Equation (3), based on the identified trade-off between decreasing the expected computational burden and controlling the variance of the weighted error. We will not consider the tuning of q any further in this work, but rather highlight that the trade-off presented here motivates the formulation and optimisation of a generalised multifidelity likelihood-free inference scheme.

3. Multifidelity inference

In Equation (3), the performance of Algorithm 1 is quantified explicitly in terms of how the Monte Carlo error between the estimate, $\hat{G}$, and the approximated posterior mean, $\mathbf{E}_{\pi_\omega}(G \mid y_0)$, decays with increasing computational budget, C_{tot}. It initially appears reasonable to conclude that the linear dependence of the performance on the expected iteration time, $\mathbf{E}(C)$, implies that if we can speed up the simulation step of Algorithm 1, then we can significantly reduce the MSE for a given computational budget.

Suppose that there exists an alternative model that we can use in Algorithm 1 in place of the original model, $f(\cdot \mid \theta)$, such that the expected computation time for each iteration, $\mathbf{E}(C)$, is significantly reduced. There are two important issues that prevent this being a viable option for improving the efficiency of likelihood-free inference. The first problem is that we need to be able to quantify the effect of the alternative model on the ratio $\mathbf{E}(W^2(G(\theta) - \mathbf{E}_{\pi_\omega}(G \mid y_0))^2)/\mathbf{E}(W)^2$ to ensure that the *overall* performance of the algorithm is improved. It is not sufficient to show that the computational burden of each iteration is reduced, since it is possible that substantially more iterations are subsequently required to achieve a specified MSE.

The second problem arises from the observation that the limiting value of $\hat{G}$, as output from Algorithm 1, is $\mathbf{E}_{\pi_\omega}(G \mid y_0)$, with residual bias,

$$\lim_{C_{\text{tot}} \to \infty} \mathbf{E}\left(\left(\hat{G} - \mathbf{E}(G \mid y_0) \right)^2 \right) = (\mathbf{E}_{\pi_\omega}(G \mid y_0) - \mathbf{E}(G \mid y_0))^2 \neq 0,$$

recalling that $\mathbf{E}_{\pi_\omega}(G \mid y_0)$ is the approximate posterior expectation induced by $L_\omega(\theta) = \mathbf{E}(\omega \mid \theta)$, and the approximand, $\mathbf{E}(G \mid y_0)$, is the posterior expectation induced by the likelihood, $\mathcal{L}(\theta) = f(y_0 \mid \theta)$. We will identify this limiting residual squared bias, $(\mathbf{E}_{\pi_\omega}(G \mid y_0) - \mathbf{E}(G \mid y_0))^2$, as the *fidelity* of the model/likelihood-free weighting pair. We emphasise here that the fidelity depends both on the model and the likelihood-free weighting used in Algorithm 1, and is contextual to the target function, G. For a given posterior mean, $\mathbf{E}(G \mid y_0)$, a model and likelihood-free weighting pair for which the value of $(\mathbf{E}_{\pi_\omega}(G \mid y_0) - \mathbf{E}(G \mid y_0))^2$ is small will be termed high-fidelity, while larger values of $(\mathbf{E}_{\pi_\omega}(G \mid y_0) - \mathbf{E}(G \mid y_0))^2$ are termed low-fidelity. Thus, if we use an alternative model in place of f in Algorithm 1, the model (and likelihood-free weighting) may be too low-fidelity, in the sense of having too large a residual squared bias versus the posterior expectation of interest, $\mathbf{E}(G \mid y_0)$.

The *multifidelity* framework overcomes both these problems, by removing the need for a binary choice between the expensive model of interest and its cheaper alternative. Instead, we carry out likelihood-free inference using information from both models. In the sections that follow, we will only consider two levels of model fidelity. However, in Section 6, we discuss possible extensions for a truly multifidelity setting with multiple approximations as our approach need not be restricted to only two models.

3.1. Multifidelity likelihood-free importance sampling

We denote the high-fidelity model and likelihood-free weighting as f_{hi} and ω_{hi}, respectively. The likelihood under the high-fidelity model is denoted $\mathcal{L}_{\text{hi}}(\theta) = f_{\text{hi}}(y_0 \mid \theta)$, and is assumed to be intractable. Following the notation introduced in Equation (1), the high-fidelity pair f_{hi} and ω_{hi} induce the approximate likelihood, $L_{\omega_{\text{hi}}}(\theta) = \mathbf{E}(\omega_{\text{hi}} \mid \theta)$ and the corresponding likelihood-free approximation to the posterior expectation, $\mathbf{E}_{\pi_{\omega_{\text{hi}}}}(G \mid y_0)$. We further assume that simulating each $\mathbf{y}_{\text{hi}} \sim f_{\text{hi}}(\cdot \mid \theta)$ is computationally expensive. This computational expense motivates the use of an approximate, low-fidelity model and likelihood-free weighting, denoted f_{lo} and ω_{lo}, respectively, inducing the approximate likelihood, $L_{\omega_{\text{lo}}}(\theta)) = \mathbf{E}(\omega_{\text{lo}} \mid \theta)$, and corresponding likelihood-free approximation to the posterior expectation, $\mathbf{E}_{\pi_{\omega_{\text{lo}}}}(G \mid y_0)$. We note that the low-fidelity model, f_{lo}, induces its own likelihood, $\mathcal{L}_{\text{lo}}(\theta) = f_{\text{lo}}(y_0 \mid \theta)$, and assume that this remains intractable, requiring a simulation-based Bayesian approach. However, we assume that simulations of the low-fidelity model, $\mathbf{y}_{\text{lo}} \sim f_{\text{lo}}(\cdot \mid \theta)$, are significantly cheaper to produce compared to simulations of the high-fidelity model, that is $\mathbf{E}(C_{\text{lo}})/\mathbf{E}(C_{\text{hi}}) \ll 1$, where C_{lo} and C_{hi} denote the random computational time of K simulations from the low-fidelity and high-fidelity model, respectively.

Given the models f_{lo} and f_{hi}, we will term the joint distribution $f_{\text{mf}}(\mathbf{y}_{\text{lo}}, \mathbf{y}_{\text{hi}} \mid \theta)$ a *multifidelity model* when f_{mf} has marginals equal to the low- and high-fidelity densities, $f_{\text{lo}}(\mathbf{y}_{\text{lo}} \mid \theta)$ and $f_{\text{hi}}(\mathbf{y}_{\text{hi}} \mid \theta)$. The models *may* be conditionally independent, such that $f_{\text{mf}}(\mathbf{y}_{\text{lo}}, \mathbf{y}_{\text{hi}} \mid \theta) = f_{\text{lo}}(\mathbf{y}_{\text{lo}} \mid \theta) f_{\text{hi}}(\mathbf{y}_{\text{hi}} \mid \theta)$, in which case simulations at each model fidelity can be carried out independently given θ. Furthermore, if the simulations are conditionally independent, this means that the resulting likelihood-free weights, $\omega_{\text{lo}}(\theta, \mathbf{y}_{\text{lo}})$ and $\omega_{\text{hi}}(\theta, \mathbf{y}_{\text{hi}})$, are also conditionally independent. However, in the more general definition of the multifidelity model as a joint distribution, we allow for *coupling* between the two fidelities. Conditioned on the low-fidelity simulations, $\mathbf{y}_{\text{lo}}$, and on parameter values, θ, we can produce a coupled simulation, $\mathbf{y}_{\text{hi}}$, from the density $f_{\text{hi}}(\mathbf{y}_{\text{hi}} \mid \theta, \mathbf{y}_{\text{lo}})$ implied by $f_{\text{mf}}(\mathbf{y}_{\text{lo}}, \mathbf{y}_{\text{hi}} \mid \theta) = f_{\text{hi}}(\mathbf{y}_{\text{hi}} \mid \theta, \mathbf{y}_{\text{lo}}) f_{\text{lo}}(\mathbf{y}_{\text{lo}} \mid \theta)$. If appropriately constructed, coupling imposes positive correlations between the resulting likelihood-free weights, $\omega_{\text{lo}}(\theta, \mathbf{y}_{\text{lo}})$ and $\omega_{\text{hi}}(\theta, \mathbf{y}_{\text{hi}})$, to enable values of ω_{lo} to provide more information about unknown values of ω_{hi}, thereby acting as a variance reduction technique [28]. Implementation of such a variance reduction approach results in a construction related to a randomised multilevel Monte Carlo scheme [15,29,30]. It should be noted, however, that the success of our multifidelity scheme does not strictly rely on finding an appropriate coupling mechanism. This is an advantage of our approach since such coupling mechanisms can be challenging to construct.

We can calculate a multifidelity likelihood-free weighting as follows. Let M be any non-negative integer-valued random variable, with conditional probability mass function $p(\cdot \mid \theta, \mathbf{y}_{\text{lo}})$, and with a positive conditional mean, $\mu(\theta, \mathbf{y}_{\text{lo}}) = \mathbf{E}(M \mid \theta, \mathbf{y}_{\text{lo}}) > 0$. Given a parameter value, θ, we define $\mathbf{z} = (\mathbf{y}_{\text{lo}}, \mathbf{y}_{\text{hi},1}, \mathbf{y}_{\text{hi},2}, \ldots, \mathbf{y}_{\text{hi},m})$, with $\mathbf{y}_{\text{lo}} \sim f_{\text{lo}}(\cdot \mid \theta)$, $m \sim p(\cdot \mid \theta, \mathbf{y}_{\text{lo}})$, and $\mathbf{y}_{\text{hi},i} \sim f_{\text{hi}}(\cdot \mid \theta, \mathbf{y}_{\text{lo}})$, noting that each $\mathbf{y}_{\text{hi},i}$ may be coupled to the low-fidelity simulation $\mathbf{y}_{\text{lo}} \sim f_{\text{lo}}(\cdot \mid \theta)$. We further define the *multifidelity likelihood-free weighting function*,

$$\omega_{\text{mf}}(\theta, \mathbf{z}) = \omega_{\text{lo}}(\theta, \mathbf{y}_{\text{lo}}) + \frac{1}{\mu(\theta, \mathbf{y}_{\text{lo}})} \sum_{i=1}^{m} \left[\omega_{\text{hi}}(\theta, \mathbf{y}_{\text{hi},i}) - \omega_{\text{lo}}(\theta, \mathbf{y}_{\text{lo}}) \right], \tag{4}$$

as the low-fidelity likelihood-free weighting, corrected by a randomly drawn number, $M = m$, of conditionally independent high-fidelity likelihood-free weightings. We write the density of $\mathbf{z}$ as $\phi(\mathbf{z} \mid \theta)$ and take expectations over $\mathbf{z}$ to obtain

$$L_{\omega_{\text{mf}}}(\theta) = \mathbf{E}(\omega_{\text{mf}} \mid \theta) = \int \omega_{\text{mf}}(\theta, \mathbf{z})\phi(\mathbf{z} \mid \theta)\, \mathrm{d}\mathbf{z} \tag{5}$$

as the multifidelity approximation to the likelihood.

Given $M = m$, only m replicates of $\mathbf{y}_{\text{hi},i} \sim f_{\text{hi}}(\cdot \mid \theta, \mathbf{y}_{\text{lo}})$ need to be simulated for $\omega_{\text{mf}}(\theta, \mathbf{z})$ to be evaluated. Thus, whenever $m = 0$, this means that no high-fidelity simulations need to be completed for $\omega_{\text{mf}}(\theta, \mathbf{z})$ to be calculated, removing the high-fidelity simulation cost from that iteration. Algorithm 2 presents the adaptation of the basic importance sampling method of Algorithm 1 to incorporate the multifidelity weighting function. The simulation step, $\mathbf{y} \sim f(\cdot \mid \theta)$, in Algorithm 1 is replaced by the MF-SIMULATE function in Algorithm 2.

Algorithm 2 Multifidelity likelihood-free importance sampling.

Require: Prior, π; importance distribution, q; likelihood-free weightings, ω_{hi} and ω_{lo}; models $f_{\text{hi}}(\cdot \mid \theta)$ and $f_{\text{lo}}(\cdot \mid \theta)$; conditional probability mass function $p(\cdot \mid \theta, \mathbf{y}_{\text{lo}})$ on non-negative integers with mean function $\mu(\theta, \mathbf{y}_{\text{lo}})$; target estimated function, G.

for $i \in [1, 2, \ldots N]$ **do**
  Sample $\theta_i \sim q(\cdot)$;
  Generate $\mathbf{z}_i \sim \phi(\cdot \mid \theta_i)$ from MF-SIMULATE(θ_i);
  For ω_{mf} in Equation (4), calculate the weight $w_i = w_{\text{mf}}(\theta, \mathbf{z}_i) = \dfrac{\pi(\theta_i)}{q(\theta_i)}\omega_{\text{mf}}(\theta_i, \mathbf{z}_i)$;
end for
Estimate expectation using weighted sum, $\hat{G}_{\text{mf}} = \dfrac{\sum_{i=1}^{N} w_i G(\theta_i)}{\sum_{j=1}^{N} w_j}$.

function MF-SIMULATE(θ)
  Simulate $\mathbf{y}_{\text{lo}} \sim f_{\text{lo}}(\cdot \mid \theta)$;
  Generate $m \sim p(\cdot \mid \theta, \mathbf{y}_{\text{lo}})$ with mean $\mu(\theta, \mathbf{y}_{\text{lo}})$;
  if $m = 0$ **then**
    return $\mathbf{z} = (\mathbf{y}_{\text{lo}})$;
  else
    for $i \in [1, 2, \ldots, m]$ **do**
      Simulate $\mathbf{y}_{\text{hi},i} \sim f_{\text{hi}}(\cdot \mid \theta, \mathbf{y}_{\text{lo}})$;
    end for
    return $\mathbf{z} = (\mathbf{y}_{\text{lo}}, \mathbf{y}_{\text{hi},1}, \ldots, \mathbf{y}_{\text{hi},m})$;
  end if
end function

3.2. Accuracy of multifidelity inference

We observe that using f_{hi} and ω_{hi} in Algorithm 1 produces an estimate of the high-fidelity approximate posterior expectation, $\mathbf{E}_{\pi_{\omega_{\text{hi}}}}(G \mid y_0)$. We show (Section 4, Proposition 3) that the multifidelity approximate likelihood, $L_{\omega_{\text{mf}}}(\theta) = \mathbf{E}(\omega_{\text{mf}}(\theta, \mathbf{z}) \mid \theta)$, is equal to the high-fidelity approximate likelihood, $L_{\omega_{\text{hi}}}(\theta) = \mathbf{E}(\omega_{\text{hi}}(\theta, \mathbf{y}_{\text{hi}}) \mid \theta)$. As a result, Algorithm 2 also produces a consistent estimate of the high-fidelity approximate posterior expectation, $\mathbf{E}_{\pi_{\omega_{\text{hi}}}}(G \mid y_0)$.

In the limit of infinite computational budgets, the estimate produced by multifidelity importance sampling in Algorithm 2 is as accurate as the estimate produced by high-fidelity importance sampling in Algorithm 1 using f_{hi} and ω_{hi}. However, we still need to show that the performance of Algorithm 2 exceeds that of Algorithm 1 in the practical context of limited computational budgets. In Section 3.3, we introduce a method to quantify the performance of Algorithms 1 and 2 and show that the performance of multifidelity inference is strongly determined by the distribution of M, the random number of high-fidelity simulations required at each iteration.

3.3. Comparing performance

Equation (3) gives the leading-order behaviour of the MSE for Algorithm 1 as the computational budget increases. A similar result applies to the output of Algorithm 2. We compare two settings: first, using Algorithm 1 with the high-fidelity model; and second, using Algorithm 2 with the multifidelity model. In the first setting we have the model, f_{hi}, and likelihood-free weighting, ω_{hi}. Each iteration has computational cost denoted C_{hi}, and produces a weighted Monte Carlo sample with weights w_i as independent draws of the random variable W_{hi}. The output of Algorithm 1 is denoted $\hat{G}_{\text{hi}}$. The MSE for Algorithm 1 has leading-order behaviour

$$\mathbf{E}\left((\hat{G}_{\text{hi}} - \mathbf{E}_{\pi_{\omega_{\text{hi}}}}(G \mid y_0))^2\right) = \left[\frac{\mathbf{E}(C_{\text{hi}})\mathbf{E}\left(W_{\text{hi}}^2(G(\theta) - \mathbf{E}_{\pi_{\omega_{\text{hi}}}}(G \mid y_0))^2\right)}{\mathbf{E}(W_{\text{hi}})^2}\right]\frac{1}{C_{\text{tot}}} + O\left(\frac{1}{C_{\text{tot}}^2}\right), \tag{6}$$

as the total simulation budget $C_{\text{tot}} \to \infty$. In the second setting, we similarly use Algorithm 2 with the multifidelity model f_{mf} and likelihood-free weighting, ω_{mf}. Each iteration has computational cost denoted C_{mf}, and produces a weighted Monte Carlo sample

with weights w_i as independent draws of the random variable W_{mf}. The output of Algorithm 2 is $\hat{G}_{\text{mf}}$. The MSE for Algorithm 2 has leading-order behaviour

$$\mathbf{E}\left((\hat{G}_{\text{mf}} - \mathbf{E}_{\pi_{\omega_{\text{mf}}}}(G \mid y_0))^2\right) = \left[\frac{\mathbf{E}(C_{\text{mf}})\mathbf{E}\left(W_{\text{mf}}^2(G(\theta) - \mathbf{E}_{\pi_{\omega_{\text{mf}}}}(G \mid y_0))^2\right)}{\mathbf{E}(W_{\text{mf}})^2}\right]\frac{1}{C_{\text{tot}}} + O\left(\frac{1}{C_{\text{tot}}^2}\right), \tag{7}$$

as the total simulation budget $C_{\text{tot}} \to \infty$. Thus the main task is to determine the conditions when

$$\mathbf{E}\left((\hat{G}_{\text{mf}} - \mathbf{E}_{\pi_{\omega_{\text{mf}}}}(G \mid y_0))^2\right) < \mathbf{E}\left((\hat{G}_{\text{hi}} - \mathbf{E}_{\pi_{\omega_{\text{hi}}}}(G \mid y_0))^2\right)$$

for the same value of C_{tot}.

The main result of the paper (Section 4, Theorem 4) provides such conditions and enables the construction of performance metrics. The performance metrics $\mathcal{J}_{\text{hi}}$ and $\mathcal{J}_{\text{mf}}[\mu]$, for Algorithm 1 and Algorithm 2, respectively, are given by

$$\mathcal{J}_{\text{hi}} = \underbrace{\mathbf{E}(C_{\text{hi}})}_{\text{high-fidelity cost}} \times \underbrace{\mathbf{E}\left(\Delta_q(\theta)^2\mathbf{E}(\omega_{\text{hi}}^2 \mid \theta)\right)}_{\text{high-fidelity variance}}, \tag{8}$$

$$\mathcal{J}_{\text{mf}}[\mu] = \underbrace{\left(\mathbf{E}(C_{\text{lo}}) + \mathbf{E}_{\rho}(\mu(\theta, \mathbf{y}_{\text{lo}})\mathbf{E}(C_{\text{hi}} \mid \theta, \mathbf{y}_{\text{lo}}))\right)}_{\text{multifidelity cost}} \times \underbrace{\left(\mathbf{E}\left(\Delta_q(\theta)^2\mathbf{E}\left(\mathbf{E}(\omega_{\text{hi}}|\theta, \mathbf{y}_{\text{lo}})^2 \mid \theta\right)\right) + \mathbf{E}_{\rho}\left(\frac{\Delta_q(\theta)^2}{\mu(\theta, \mathbf{y}_{\text{lo}})}\mathbf{E}\left(\left(\omega_{\text{hi}} - \omega_{\text{lo}}\right)^2 \mid \theta, \mathbf{y}_{\text{lo}}\right)\right)\right)}_{\text{multifidelity variance}}, \tag{9}$$

where M is a Poisson distributed random variable with mean $\mu(\theta, \mathbf{y}_{\text{lo}})$, $\Delta_q(\theta) = \left(G(\theta) - \mathbf{E}_{\pi_{\omega_{\text{hi}}}}(G \mid y_0)\right)\pi(\theta)/q(\theta)$ is the importance weighted error, $\rho(\theta, \mathbf{y}_{\text{lo}}) = f_{\text{lo}}(\mathbf{y}_{\text{lo}} \mid \theta)q(\theta)$ is the joint density of the low-fidelity simulator and the parameters under the importance distribution. Effectively both metrics are reformulations of the numerators of the leading order terms in Equation (6) and Equation (7), respectively.

The condition $\mathcal{J}_{\text{mf}}[\mu] < \mathcal{J}_{\text{hi}}$ is shown to be a necessary and sufficient condition for Algorithm 2 out-performing Algorithm 1 for the same computational budget (Theorem 4). Importantly, the performance depends explicitly on the free choice of the function $\mu(\theta, \mathbf{y}_{\text{lo}})$ that determines the conditional mean of the Poisson-distributed number of high-fidelity simulations required at each iteration. We observe from the first factor in Equation (9) that, when μ is smaller, the total simulation cost is less. However, the second factor of Equation (9) implies that as μ decreases, the variability of the likelihood-free weighting can increase without bound, which can severely damage the performance. Thus, Equation (9) illustrates the characteristic multifidelity trade-off between reducing simulation burden while also controlling the increase in sample variance. In the results that follow, we continue to assume M is a Poisson distributed random variable, however, considerations for binomial and geometric distributions are given in Appendix B.

Using classical results from calculus of variations, it is possible to determine the mean function, $\mu^\star$, that achieves optimal performance of Algorithm 2, in the sense of minimising the functional, $\mathcal{J}_{\text{mf}}[\mu]$. This optimal function, $\mu^\star$, that maximises performance on average is given by

$$\mu^\star(\theta, \mathbf{y}_{\text{lo}})^2 = \frac{\mathbf{E}(C_{\text{lo}})\Delta_q(\theta)^2\mathbf{E}\left(\left(\omega_{\text{hi}} - \omega_{\text{lo}}\right)^2 \mid \theta, \mathbf{y}_{\text{lo}}\right)}{\mathbf{E}(C_{\text{hi}} \mid \theta, \mathbf{y}_{\text{lo}})\mathbf{E}\left(\Delta_q(\theta)^2\mathbf{E}\left(\mathbf{E}(\omega_{\text{hi}}|\theta, \mathbf{y}_{\text{lo}})^2 \mid \theta\right)\right)}. \tag{10}$$

This result is derived in Section 4 (Lemma 5). Note that as $\mathbf{E}\left(\left(\omega_{\text{hi}} - \omega_{\text{lo}}\right)^2 \mid \theta, \mathbf{y}_{\text{lo}}\right)$ increases then $\mu^\star$ increases. This means that if the expected error between the likelihood-free weightings is large, then the high-fidelity model needs to be simulated more frequently to correct for this. Conversely, when $\mathbf{E}(C_{\text{hi}} \mid \theta, \mathbf{y}_{\text{lo}})$ is larger, the greater simulation time of the high-fidelity model means that $\mu^\star$ should be smaller, and the requirement for the most expensive simulations is reduced. Intuitively, $\mu^\star$ acts to balance the trade-off between controlling simulation cost and variance identified in Equation (9) above.

Of course, it need not be the case that the optimal mean function in Equation (10) leads to $\mathcal{J}_{\text{mf}}[\mu] < \mathcal{J}_{\text{hi}}$. For this to occur we require (See Corollary 6),

$$\left(\frac{\mathbf{E}(C_{\text{lo}})}{\mathbf{E}(C_{\text{hi}})}\frac{\mathbf{E}\left(\Delta_q(\theta)^2\mathbf{E}\left(\mathbf{E}(\omega_{\text{hi}}|\theta, \mathbf{y}_{\text{lo}})^2 \mid \theta\right)\right)}{\mathbf{E}\left(\Delta_q(\theta)^2\mathbf{E}(\omega_{\text{hi}}^2 \mid \theta)\right)}\right)^{1/2} + \mathbf{E}_{\rho}\left(\left(\frac{\Delta_q(\theta)^2\mathbf{E}\left(\left(\omega_{\text{hi}} - \omega_{\text{lo}}\right)^2 \mid \theta, \mathbf{y}_{\text{lo}}\right)}{\mathbf{E}\left(\Delta_q(\theta)^2\mathbf{E}(\omega_{\text{hi}}^2 \mid \theta)\right)}\frac{\mathbf{E}(C_{\text{hi}} \mid \theta, \mathbf{y}_{\text{lo}})}{\mathbf{E}(C_{\text{hi}})}\right)^{1/2}\right) < 1. \tag{11}$$

The first term in Equation (11) justifies the key assumption that the average computational cost of the low-fidelity model is as small as possible compared to that of the high-fidelity model, $\mathbf{E}(C_{\text{lo}})/\mathbf{E}(C_{\text{hi}}) \ll 1$. The second term is a measure of the total detriment to the performance of Algorithm 2 incurred by the inaccuracy of ω_{lo} versus ω_{hi} as a Monte Carlo estimate of $L_{\omega_{\text{hi}}}$, as quantified by $\mathbf{E}\left(\left(\omega_{\text{hi}} - \omega_{\text{lo}}\right)^2 \mid \theta, \mathbf{y}_{\text{lo}}\right)$. This condition justifies two key criteria for the success of the multifidelity method: that low-fidelity simulations are significantly cheaper than high-fidelity simulations, and that the likelihood-free weightings, ω_{hi} and ω_{lo}, agree

sufficiently well, on average. More detailed analysis of necessary and sufficient conditions for the existence of $\mu^\star$ is given in Section 4, following the proof of Corollary 6.

These key results show not only correctness of the multifidelity approach asymptotically, but also the situations in which performance improvement is expected. The details of the analysis results can be found in the proofs of Theorem 4, Lemma 5 and Corollary 6 given in Section 4. However, these analytical results are only useful insofar as the various expectations given in Equations (8) and (9) are known. Typically these expectations will be unknown *a priori*, and need to be estimated. In the following section, we describe how the analytical results of Section 3.3 can be used to construct a heuristic for adaptive multifidelity inference that learns a near-optimal mean function, $\mu(\theta, \mathbf{y}_{\text{lo}})$, as simulations at each fidelity are completed.

3.4. Practical implementation

In Section 3.3, we derived the optimal mean function for the Poisson distribution of the number of high-fidelity simulations, M, generated in an iteration of Algorithm 2, conditioned on the parameter value, θ, and low-fidelity simulation, $\mathbf{y}_{\text{lo}}$. In this section, we describe a practical approach to determining a near-optimal mean function for use in multifidelity likelihood-free inference. We rely on two approximations, relative to the analytically optimal mean function $\mu^\star$ given in Equation (10). First, we constrain the optimisation of $\mathcal{J}_{\text{mf}}$ to the space of functions $\mu_{\mathcal{D}}$ that are piecewise constant in an arbitrary, given, finite partition, $\mathcal{D}$, of the global space of $(\theta, \mathbf{y}_{\text{lo}})$ values. The resulting optimisation problem is therefore finite-dimensional. However, although this optimisation can be solved analytically, we can observe that its estimation, being based on the ratios of simulation-based Monte Carlo estimates, is numerically unstable. This motivates a second approximation, which is to follow a gradient-descent approach to allow the mean function to adaptively converge towards the optimum.

We constrain the space of mean functions, μ, to be piecewise constant. Consider an arbitrary, given collection $\mathcal{D} = \{D_k \mid k = 1, \ldots, K\}$ of ρ-integrable sets that partition the global space of $(\theta, \mathbf{y}_{\text{lo}})$ values. We denote a $\mathcal{D}$-piecewise constant function, parameterised by the vector $\nu = (\nu_1, \ldots, \nu_k)$, as

$$\mu_{\mathcal{D}}(\theta, \mathbf{y}_{\text{lo}};\ \nu) = \sum_{k=1}^{K} \nu_k \mathbf{I}((\theta, \mathbf{y}_{\text{lo}}) \in D_k).$$

Substituting this function into Equation (9), we can quantify the performance of Algorithm 2, using the mean defined by $\mu_{\mathcal{D}}(\theta, \mathbf{y}_{\text{lo}};\ \nu)$, as the parameterised product

$$\mathcal{J}_{\mathcal{D}}(\nu) = \left(\mathbf{E}(C_{\text{lo}}) + \sum_{k=1}^{K} \nu_k C_k\right)\left(\mathbf{E}\left(\Delta_q(\theta)^2 \mathbf{E}\left(\mathbf{E}(\omega_{\text{hi}} \mid \theta, \mathbf{y}_{\text{lo}})^2 \mid \theta\right)\right) + \sum_{k=1}^{K} \frac{V_k}{\nu_k}\right), \tag{12}$$

where, for convenience in this derivation, we denote

$$C_k = \mathbf{E}_\rho\left(\mathbf{E}(C_{\text{hi}} \mid \theta, \mathbf{y}_{\text{lo}}) \mathbf{I}((\theta, \mathbf{y}_{\text{lo}}) \in D_k)\right),$$
$$V_k = \mathbf{E}_\rho\left(\Delta_q(\theta)^2 \mathbf{E}\left(\left(\omega_{\text{hi}} - \omega_{\text{lo}}\right)^2 \mid \theta, \mathbf{y}_{\text{lo}}\right) \mathbf{I}((\theta, \mathbf{y}_{\text{lo}}) \in D_k)\right),$$

for $k = 1, 2, \ldots, K$.

Just as in the case of general μ^*, we can optimise the function $\mathcal{J}_{\mathcal{D}}(\nu)$ across positive vectors, ν, to obtain the result,

$$\nu_k^\star = \sqrt{\frac{V_k \mathbf{E}(C_{\text{lo}})}{C_k \mathbf{E}\left(\Delta_q(\theta)^2 \mathbf{E}\left(\mathbf{E}(\omega_{\text{hi}} \mid \theta, \mathbf{y}_{\text{lo}})^2 \mid \theta\right)\right)}}, \quad \text{for } k = 1, 2, \ldots, K. \tag{13}$$

This result is more convenient since all the terms on the right-hand side are constants. However, we still need to estimate these expectations values as they are unknown *a priori*. The standard approach would typically be to perform trial samples to estimate these values with Monte Carlo. This could be problematic in practice due to the rational form of $\nu_k^\star$ which means that these estimates can be unstable, particularly for sets $D_k \in \mathcal{D}$ with small volume under ρ.

We now consider a conservative approach to determining values for ν that will provide stable estimates of $\nu^\star$. Rather than directly targeting $\nu^\star$, based on ratios of highly variable Monte Carlo estimates, we can introduce a gradient-descent approach to updating the vector ν. Taking derivatives of $\mathcal{J}_{\mathcal{D}}$ with respect to $\log \nu_k$ for $k = 1, \ldots, K$ gives the gradient

$$\frac{\partial \mathcal{J}_{\mathcal{D}}}{\partial \log \nu_k} = \nu_k C_k \left(\mathbf{E}\left(\Delta_q(\theta)^2 \mathbf{E}\left(\mathbf{E}(\omega_{\text{hi}} \mid \theta, \mathbf{y}_{\text{lo}})^2 \mid \theta\right)\right) + \sum_{j=1}^{K} \frac{V_j}{\nu_j}\right) - \frac{V_k}{\nu_k}\left(\mathbf{E}(C_{\text{lo}}) + \sum_{j=1}^{K} C_j \nu_j\right).$$

Thus, if we write $\nu^{(r)}$ for the value of ν used in iteration r of Algorithm 2, we intend to update to $\nu^{(r+1)}$ in the next iteration using gradient descent, such that

$$\log \nu_k^{(r+1)} = \log \nu_k^{(r)} - \delta\left[\nu_k^{(r)} C_k \left(\mathbf{E}\left(\Delta_q(\theta)^2 \mathbf{E}\left(\mathbf{E}(\omega_{\text{hi}} \mid \theta, \mathbf{y}_{\text{lo}})^2 \mid \theta\right)\right) + \sum_{j=1}^{K} \frac{V_j}{\nu_j^{(r)}}\right) - \frac{V_k}{\nu_k^{(r)}}\left(\mathbf{E}(C_{\text{lo}}) + \sum_{j=1}^{K} c_j \nu_j^{(r)}\right)\right]. \tag{14}$$

Note that we express this updating rule in terms of $\log \nu_k^{(r)}$ to ensure that each $\nu_k^{(r)}$ is positive, since the updates to $\nu^{(r)}$ are multiplicative. As is typical of gradient-descent approaches, Equation (14) requires the specification of the step size hyperparameter, δ.

It is straightforward to show that $\nu^\star$ is the unique positive stationary point of Equation (14). Furthermore, since each derivative $\partial \mathcal{J}_{\mathcal{D}}/\partial \log \nu_k$ is quadratic in the expectations the numerical instability in estimating Equation (13) as a ratio does not occur. In relatively under-sampled regions $D_k \in \mathcal{D}$ with small ρ-volume, the small values of c_k and V_k ensure that the convergence to the corresponding estimated optimal value, $\nu_k^\star$, is more conservative.

We now explicitly set out the Monte Carlo estimates of unknown expectations given r-iterations of the multifidelity inference scheme (Algorithm 2). Firstly, the cost related expectations are estimated using

$$\mathbf{E}(C_{\text{lo}}) \approx \hat{C}_{\text{lo}}^{(r)} = \frac{1}{r}\sum_{i=1}^{r} c_{\text{lo},i}, \tag{15}$$

and for $k = 1, 2, \ldots, K$,

$$C_k \approx \hat{C}_k^{(r)} = \frac{1}{r}\sum_{i=1}^{r} \mathbf{I}((\theta_i, \mathbf{y}_{\text{lo},i}) \in D_k)\frac{1}{\mu_i}\sum_{j=1}^{m_i} c_{\text{hi},i,j}, \tag{16}$$

where, for iteration i, $c_{\text{lo},i}$ is the observed simulation cost of each $\mathbf{y}_{\text{lo},i}$ given parameters θ_i drawn from $q(\theta)$, and $c_{\text{hi},i,j}$ is the observed simulation cost of each $\mathbf{y}_{\text{hi},i,j}$ for $j = 1, \ldots, m_i$ with m_i as a random draw from the distribution of M having mean μ_i. The estimators for the variance related terms are more complex with

$$\mathbf{E}\left(\Delta_q(\theta)^2 \mathbf{E}\left(\mathbf{E}(\omega_{\text{hi}}|\theta, \mathbf{y}_{\text{lo}})^2 \mid \theta\right)\right) \approx \hat{V}_{\text{mf}}^{(r)} = \frac{1}{r}\sum_{i=1}^{r}\left(\frac{\Delta_i^{(r)}}{\mu_i}\right)^2\left[\left(\sum_{j=1}^{m_i}\omega_{\text{hi},i,j}\right)^2 - \sum_{j=1}^{m_i}\omega_{\text{hi},i,j}^2\right], \tag{17}$$

and for $k = 1, 2, \ldots, K$,

$$V_k \approx \hat{V}_k^{(r)} = \frac{1}{r}\sum_{i=1}^{r}\mathbf{I}_{D_k}(\theta_i, \mathbf{y}_{\text{lo},i})\frac{1}{\mu_i}\sum_{j=1}^{m_i}\left(\Delta_i^{(r)}\left(\omega_{\text{hi},i,j} - \omega_{\text{lo},i}\right)\right)^2, \tag{18}$$

where $\omega_{\text{lo},i} = \omega_{\text{lo}}(\theta_i, \mathbf{y}_{\text{lo},i})$, $\omega_{\text{hi},i,j} = \omega_{\text{hi}}(\theta_i, \mathbf{y}_{\text{hi},i,j})$ for $j = 1, \ldots, m_i$, $\Delta_i^{(r)} = \left[G(\theta_i) - \hat{G}_{\text{hi}}^{(r)}\right]\pi(\theta_i)/q(\theta_i)$ with $\hat{G}_{\text{hi}}^{(r)}$ as the current Monte Carlo estimate of the target $\mathbf{E}_{\pi_{\omega_{\text{hi}}}}(G \mid y_0)$.

These estimates can then be substituted into Equation (14) to produce an updating rule for $\nu_k^{(r)}$, that is,

$$\log \nu_k^{(r+1)} = \log \nu_k^{(r)} - \delta\left[\nu_k^{(r)} c_k^{(r)}\left(V_{\text{mf}}^{(r)} + \sum_{j=1}^{K}\frac{V_j^{(r)}}{\nu_j^{(r)}}\right) - \frac{V_k^{(r)}}{\nu_k^{(r)}}\left(\hat{C}_{\text{lo}}^{(r)} + \sum_{j=1}^{K} c_j^{(r)}\nu_j^{(r)}\right)\right], \tag{19}$$

leading to an adaptive multifidelity likelihood-free importance sampling scheme, as outlined in Algorithm 3.

In addition to the specification of the step size hyperparameter, δ, Algorithm 3 also requires a burn-in phase, N_0, to initialise the Monte Carlo estimates in Equation (15)–(18). The partition, $\mathcal{D} = \{D_1, \ldots, D_K\}$, is also an input into Algorithm 3. We defer an investigation of how to choose this partition to future work. For the purposes of this paper, however, we can heuristically construct a partition, $\mathcal{D}$, by fitting a decision tree. We use the burn-in phase of Algorithm 3, over iterations $i \leq N_0$, and regress the values of

$$\mu_i^\star = \left|\Delta_i^{(N_0)}\right|\sqrt{\frac{\sum_j(\omega_{\text{hi},i,j} - \omega_{\text{lo},i})^2}{\sum_j c_{\text{hi},i,j}}},$$

against features $(\theta_i, \mathbf{y}_{\text{lo},i})$, using the CART algorithm [31] as implemented in `DecisionTrees.jl`. Note that this regression is motivated by the form of the true optimal mean function, $\mu^\star$, given in Equation (10). The resulting decision tree defines a partition, $\mathcal{D} = \{D_1, \ldots, D_K\}$, used to define the piecewise-constant mean function $\mu_{\mathcal{D}}(\theta, y_{\text{lo}};\ \nu)$ over $i > N_0$.

4. Analysis of likelihood-free and multifidelity importance sampling

In this section, we detail the theoretical analysis of the general likelihood-free importance sampling scheme and the multifidelity schemes. The results in this section are referred to throughout Sections 2 and 3, however, notation in this section may be distinct to ensure the analysis is not needlessly verbose. Throughout the relation $f(x) \preccurlyeq g(x)$ is taken to mean *there exists a constant, c, such that $f(x) \leq cg(x)$ as $x \to \infty$*. Likewise $f(x) \prec g(x)$ is taken to mean *there exists a constant, c, such that $f(x) < cg(x)$ as $x \to \infty$*.

The first result in Theorem 1 establishes the consistency of the general likelihood-free importance sampling scheme (Algorithm 1) and in doing so derives the convergence rate in MSE. For notational simplicity, we define the function $\Delta(\theta) = G(\theta) - \mathbf{E}_{\pi_\omega}(G \mid y_0)$ to recentre the approximate posterior mean.

Theorem 1. *For the weighted sample values (θ_i, w_i) produced in each iteration of Algorithm 1, let W denote the random value of the weight w_i, and let θ denote the random value of θ_i. The mean squared error (MSE) of the estimator, $\hat{G}$, is given to leading order by*

Algorithm 3 Adaptive multifidelity likelihood-free importance sampling.

Require: Prior, π; importance distribution, q; likelihood-free weightings, ω_{hi} and ω_{lo}; models $f_{\text{hi}}(\cdot \mid \theta)$ and $f_{\text{lo}}(\cdot \mid \theta)$; partition $\mathcal{D} = \{D_1, \ldots, D_K\}$ of $(\theta, \mathbf{y}_{\text{lo}})$ space; adaptation rate, δ; burn-in period, N_0; target estimated function, G.

Initialise $\log \nu_k^{(1)} = 0$ for $k = 1, \ldots, K$.
for $i \in [1, 2, \ldots, N]$ **do**
  Sample $\theta_i \sim q(\cdot)$;
  Generate $\mathbf{z}_i \sim \phi(\cdot \mid \theta_i)$ from MF-SIMULATE$(\theta_i, \nu^{(i)})$;
  For ω_{mf} in Equation (4), calculate the weight $w_i = w_{\text{mf}}(\theta, \mathbf{z}_i) = \dfrac{\pi(\theta_i)}{q(\theta_i)} \omega_{\text{mf}}(\theta_i, \mathbf{z}_i)$;
  if $i > N_0$ **then**
    Generate $\nu^{(i+1)}$ from UPDATE-NU$(\nu^{(i)})$;
  else
    Set $\nu^{(i+1)} = \nu^{(i)}$;
  end if
end for
Estimate expectation using weighted sum, $\hat{G}_{\text{mf}} = \dfrac{\sum_{i=1}^N w_i G(\theta_i)}{\sum_{j=1}^N w_j}$.
function MF-SIMULATE(θ, ν)
  Simulate $\mathbf{y}_{\text{lo}} \sim f_{\text{lo}}(\cdot \mid \theta)$;
  Find k such that $(\theta, \mathbf{y}_{\text{lo}}) \in D_k$;
  Generate $m \sim \text{Poi}(\nu_k)$;
  for $i = 1, \ldots, m$ **do**
    Simulate $\mathbf{y}_{\text{hi},i} \sim f_{\text{hi}}(\cdot \mid \theta, \mathbf{y}_{\text{lo}})$;
  end for
  return $\mathbf{z} = (\mathbf{y}_{\text{lo}}, m, \mathbf{y}_{\text{hi},1}, \ldots, \mathbf{y}_{\text{hi},m})$
end function

function UPDATE-NU$(\nu_1, \ldots, \nu_K)$
  Update Monte Carlo estimates defined in Equation (15)–(18);
  for $k = 1, \ldots, K$ **do**
    Increment $\log \nu_k$ according to Equation (19);
  end for
  return $\nu = (\nu_1, \ldots, \nu_K)$
end function

$$\mathbf{E}\left((\hat{G} - \mathbf{E}_{\pi_\omega}(G \mid y_0))^2\right) \leqslant \left[\frac{\mathbf{E}\left(W^2\Delta^2\right)}{\mathbf{E}(W)^2}\right] \frac{1}{N},$$

as $N \to \infty$. Thus, $\hat{G}$ is a consistent estimator of $\mathbf{E}_{\pi_\omega}(G \mid y_0)$.

Proof. The Monte Carlo estimate produced by Algorithm 1, $\hat{G} = R/S$, is the ratio of two random variables: the weighted sum, $R = \sum_{i=1}^N w(\theta_i, \mathbf{y}_i) G(\theta_i)$, and the normalising sum, $S = \sum_{i=1}^N w(\theta_i, \mathbf{y}_i)$. We write the function $\Phi(r, s) = (r/s - \mathbf{E}_{\pi_\omega}(G \mid y_0))^2$, and note that the MSE is the expected value of the function $(\hat{G} - \mathbf{E}_{\pi_\omega}(G \mid y_0))^2 = \Phi(R, S)$. Using the delta method, we take expectations of the second-order Taylor expansion of $\Phi(R, S)$ about $(\mu_R, \mu_S) = (\mathbf{E}(R), \mathbf{E}(S))$, to give

$$\mathbf{E}\left(\Phi(R,S)\right) = \Phi(\mu_R, \mu_S) + \frac{1}{\mu_S^2}\left[\text{Var}(R) + \left(2\mathbf{E}_{\pi_\omega}(G \mid y_0) - \frac{4\mu_R}{\mu_S}\right)\text{Cov}(R,S) + \left(\frac{3\mu_R - 2\mathbf{E}_{\pi_\omega}(G \mid y_0)\mu_S}{\mu_S^2}\right)\mu_R \text{Var}(S)\right]$$
$$+ O\left(\frac{\mathbf{E}\left(\left(\left(R - \mu_R\right) + \left(S - \mu_S\right)\right)^3\right)}{\mu_S^3}\right).$$

Taking expectations with respect to N independent draws of $(\theta, \mathbf{y})$ with density $f(\mathbf{y} \mid \theta) q(\theta)$, it is straightforward to write

$$\mu_R = \mathbf{E}(R) = N\mathbf{E}(WG) = N Z_\omega \mathbf{E}_{\pi_\omega}(G \mid y_0),$$

$$\mu_S = \mathbf{E}(S) = N\mathbf{E}(W) = N Z_\omega,$$

where we recall that $\mathbf{E}(W) = Z_\omega = \int L_\omega(\theta)\pi(\theta)\, \mathrm{d}\theta$ is the normalising constant in Section 2.1. We substitute these expectations into the Taylor expansion of $\mathbf{E}((\hat{G} - \mathbf{E}_{\pi_\omega}(G \mid y_0))^2)$, noting that the leading-order term, $\Phi(\mu_R, \mu_S)$, is zero. Thus, we can write the dominant behaviour of the MSE as

$$\mathbf{E}\left((\hat{G}-\mathbf{E}_{\pi_\omega}(G\mid y_0))^2\right)=\frac{1}{N^2Z_\omega^2}\left[\mathrm{Var}(R)-2\mathbf{E}_{\pi_\omega}(G\mid y_0)\mathrm{Cov}(R,S)+\mathbf{E}_{\pi_\omega}(G\mid y_0)^2\mathrm{Var}(S)\right]+O\left(\frac{\mathbf{E}\left(\left((R-\mu_R)+(S-\mu_S)\right)^3\right)}{N^3}\right)$$

$$=\frac{1}{N^2Z_\omega^2}\mathrm{Var}(R-\mathbf{E}_{\pi_\omega}(G\mid y_0)S)+O\left(\frac{\mathbf{E}\left(\left((R-\mu_R)+(S-\mu_S)\right)^3\right)}{N^3}\right),$$

as $N\to\infty$. Substituting into this expression the definitions of R and S as summations of N independent identically distributed random variables, we have

$$\mathbf{E}((\hat{G}-\mathbf{E}_{\pi_\omega}(G\mid y_0))^2)=\frac{1}{N^2Z_\omega^2}N\mathrm{Var}(W\Delta)+O\left(\frac{N}{N^3}\right),$$

and Equation (2) follows, on noting that $\mathbf{E}(W\Delta)=0$ and that $\mathbf{E}(W)=Z_\omega$. □

The consistency and MSE convergence rate in terms of sample size N is trivially recast in Corollary 2 in terms of a computational budget C_{tot}. This form is particularly important in the context of performance comparison with the multifidelity schemes as presented later.

Corollary 2. *Let the computational cost of each iteration of Algorithm 1 be denoted by the random variable C. The leading order behaviour of the MSE of $\hat{G}$ as an estimate of $\mathbf{E}_{\pi_\omega}(G\mid y_0)$ is*

$$\mathbf{E}\left((\hat{G}-\mathbf{E}_{\pi_\omega}(G\mid y_0))^2\right)\lesssim\left[\frac{\mathbf{E}(C)\mathbf{E}\left(W^2\Delta^2\right)}{\mathbf{E}(W)^2}\right]\frac{1}{C_{\text{tot}}},$$

as $C_{\text{tot}}\to\infty$.

Proof. As the given computational budget increases, $C_{\text{tot}}\to\infty$, the Monte Carlo sample size that can be produced with that budget increases on the order of $N\sim C_{\text{tot}}/\mathbf{E}(C)$. On substituting this expression into Equation (2), the result follows. □

Given the multifidelity scheme in Algorithm 2 is also a special case of Algorithm 1, we know via Theorem 1 that the estimator is consistent with respect to $\mathbf{E}_{\pi_{\omega_{\text{mf}}}}(G\mid y_0)$. Proposition 3 further establishes that in the multifidelity case we also have $\mathbf{E}_{\pi_{\omega_{\text{mf}}}}(G\mid y_0)=\mathbf{E}_{\pi_{\omega_{\text{hi}}}}(G\mid y_0)$.

Proposition 3. *The multifidelity approximation to the likelihood, $L_{\text{mf}}(\theta)=\mathbf{E}(\omega_{\text{mf}}\mid\theta)$, is equal to the high-fidelity approximation to the likelihood, $L_{\text{hi}}(\theta)=\mathbf{E}(\omega_{\text{hi}}\mid\theta)$. Therefore, the estimate $\hat{G}_{\text{mf}}$ produced by Algorithm 2 is a consistent estimate of the high-fidelity approximate posterior expectation, $\mathbf{E}_{\pi_{\omega_{\text{hi}}}}(G\mid y_0)$.*

Proof. We take the expectation of ω_{mf} (Equation (4)) conditional on $(\theta,\mathbf{y}_{\text{lo}},m)$, to find

$$\mathbf{E}(\omega_{\text{mf}}\mid\theta,\mathbf{y}_{\text{lo}},M=m)=\left(1-\frac{m}{\mu}\right)\omega_{\text{lo}}(\theta,\mathbf{y}_{\text{lo}})+\frac{m}{\mu}\mathbf{E}(\omega_{\text{hi}}\mid\theta,\mathbf{y}_{\text{lo}}).$$

Further taking expectations over the random integer M, which has conditional expected value $\mu(\theta,\mathbf{y}_{\text{lo}})$, gives

$$\mathbf{E}(\omega_{\text{mf}}\mid\theta,\mathbf{y}_{\text{lo}})=\mathbf{E}(\omega_{\text{hi}}\mid\theta,\mathbf{y}_{\text{lo}}).$$

Further taking expectations with respect to $\mathbf{y}_{\text{lo}}$, it follows that $L_{\omega_{\text{mf}}}(\theta)=L_{\omega_{\text{hi}}}(\theta)$. Therefore, the likelihood-free approximate posteriors, $\pi_{\omega_{\text{mf}}}=\pi_{\omega_{\text{hi}}}$, are equal and thus $\hat{G}_{\text{mf}}$ is a consistent estimate of $\mathbf{E}_{\pi_{\omega_{\text{mf}}}}(G\mid y_0)=\mathbf{E}_{\pi_{\omega_{\text{hi}}}}(G\mid y_0)$, as required. □

We now arrive at the most important result in this work as it determines the necessary and sufficient conditions for which the multifidelity scheme out-performs direct likelihood-free importance sampling targeting the high-fidelity weighting. For concreteness, Theorem 4 assumes that the random variable M determining the required number of high-fidelity simulations in each iteration of Algorithm 2, is Poisson distributed, conditional on the parameter value and low-fidelity simulation output. However, it is important to note that only minor alterations to the form of the performance metrics arise through other parametric assumptions on M. Binomial and Geometric distribution alternatives are explored in Appendix B. We show that, for a given multifidelity model and likelihood-free weightings, the mean function, μ, for M determines the performance of Algorithm 2 relative to Algorithm 1.

Theorem 4. *Assume that the random number of high-fidelity simulations, M, required in each iteration of Algorithm 2 is Poisson distributed with conditional mean $\mu(\theta,\mathbf{y}_{\text{lo}})$. Let $c_{\text{hi}}(\theta)$ [respectively, $c_{\text{lo}}(\theta)$ and $c_{\text{hi}}(\theta,\mathbf{y}_{\text{lo}})$] be the expected time taken to simulate $\mathbf{y}_{\text{hi}}\sim f_{\text{hi}}(\cdot\mid\theta)$ [respectively, to simulate $\mathbf{y}_{\text{lo}}\sim f_{\text{lo}}(\cdot\mid\theta)$ and to produce the coupled high-fidelity simulation $\mathbf{y}_{\text{hi}}\sim f_{\text{hi}}(\cdot\mid\theta,\mathbf{y}_{\text{lo}})$]. Further, assume that the computational*

cost of each iteration of Algorithm 1 and Algorithm 2 can be approximated by the dominant cost of simulation alone, neglecting the costs of the other calculations.

As $C_{\text{tot}} \to \infty$, the MSE of the high-fidelity likelihood-free importance sampling estimator, $\hat{G}_{\omega_{\text{hi}}}$, asymptotically exceeds that of the multifidelity importance sampling estimator, $\hat{G}_{\omega_{\text{mf}}}$, that is,

$$\mathbf{E}\left((\hat{G}_{\omega_{\text{mf}}} - \mathbf{E}_{\pi_{\omega_{\text{hi}}}}(G \mid y_0))^2\right) < \mathbf{E}\left((\hat{G}_{\omega_{\text{hi}}} - \mathbf{E}_{\pi_{\omega_{\text{hi}}}}(G \mid y_0))^2\right),$$

if and only if $\mathcal{J}_{\text{mf}}[\mu] < \mathcal{J}_{\text{hi}}$, where

$$\mathcal{J}_{\text{hi}} = c_{\text{hi}} V_{\text{hi}}, \tag{20a}$$

$$\mathcal{J}_{\text{mf}}[\mu] = \left(c_{\text{lo}} + \mathbf{E}_{\rho}\left(\mu(\theta, \mathbf{y}_{\text{lo}}) c_{\text{hi}}(\theta, \mathbf{y}_{\text{lo}})\right)\right) \times \left(V_{\text{mf}} + \mathbf{E}_{\rho}\left(\Delta_q(\theta)^2 \frac{\eta(\theta, \mathbf{y}_{\text{lo}})}{\mu(\theta, \mathbf{y}_{\text{lo}})}\right)\right), \tag{20b}$$

with constants

$$c_{\text{hi}} = \int c_{\text{hi}}(\theta)\, q(\theta) \mathrm{d}\theta, \quad V_{\text{hi}} = \int \Delta_q(\theta)^2 \mathbf{E}(\omega_{\text{hi}}^2 \mid \theta)\, q(\theta) \mathrm{d}\theta, \tag{20c}$$

$$c_{\text{lo}} = \int c_{\text{lo}}(\theta)\, q(\theta) \mathrm{d}\theta, \quad V_{\text{mf}} = \int \Delta_q(\theta)^2 \mathbf{E}(\lambda_{\text{hi}}^2 \mid \theta)\, q(\theta) \mathrm{d}\theta, \tag{20d}$$

and functions

$$\Delta_q(\theta) = \frac{\pi(\theta)}{q(\theta)}\left(G(\theta) - G_{\text{hi}}\right), \quad \eta(\theta, \mathbf{y}_{\text{lo}}) = \mathbf{E}\left(\left(\omega_{\text{hi}} - \omega_{\text{lo}}\right)^2 \mid \theta, \mathbf{y}_{\text{lo}}\right), \quad \lambda_{\text{hi}}(\theta, \mathbf{y}_{\text{lo}}) = \mathbf{E}(\omega_{\text{hi}} \mid \theta, \mathbf{y}_{\text{lo}}), \tag{20e}$$

and the joint density

$$\rho(\theta, \mathbf{y}_{\text{lo}}) = f_{\text{lo}}(\mathbf{y}_{\text{lo}} \mid \theta) q(\theta). \tag{20f}$$

Proof. The leading order performance of each of Algorithm 1 and Algorithm 2 is given in terms of increasing computational budget, C_{tot}, in Equation (6) and Equation (7), respectively. For the performance of Algorithm 2 to exceed that of Algorithm 1, we compare the leading order coefficients from Equations (6) and (7), requiring

$$\frac{\mathbf{E}(C_{\text{mf}})\mathbf{E}(w_{\text{mf}}^2 \Delta^2)}{\mathbf{E}(w_{\text{mf}})^2} < \frac{\mathbf{E}(C_{\text{hi}})\mathbf{E}(w_{\text{hi}}^2 \Delta^2)}{\mathbf{E}(w_{\text{hi}})^2}. \tag{21}$$

We note that $\mathbf{E}(w_{\text{mf}} \mid \theta) = \pi(\theta) L_{\omega_{\text{mf}}}(\theta)/q(\theta)$ and $\mathbf{E}(w_{\text{hi}} \mid \theta) = \pi(\theta) L_{\omega_{\text{hi}}}(\theta)/q(\theta)$. Since $L_{\text{mf}} = L_{\text{hi}}$, as shown in Proposition 3, the denominators in Equation (21) are therefore equal. Thus,

$$\mathcal{J}_{\text{mf}} = \mathbf{E}(C_{\text{mf}})\mathbf{E}(w_{\text{mf}}^2 \Delta^2) < \mathbf{E}(C_{\text{hi}})\mathbf{E}(w_{\text{hi}}^2 \Delta^2) = \mathcal{J}_{\text{hi}},$$

is the condition for Algorithm 2 to outperform Algorithm 1.

Taking the right-hand side of this inequality first, clearly the expected simulation time is $\mathbf{E}(C_{\text{hi}}) = c_{\text{hi}}$, for the constant c_{hi} defined in Equation (20c). Similarly, we can write

$$\mathbf{E}(w_{\text{hi}}^2 \Delta^2) = \int \left(\frac{\pi(\theta)}{q(\theta)} \Delta(\theta)\right)^2 \left[\int \omega_{\text{hi}}(\theta, \mathbf{y}_{\text{hi}})^2 f_{\text{hi}}(\mathbf{y}_{\text{hi}} \mid \theta) \mathrm{d}\mathbf{y}_{\text{hi}}\right] q(\theta) \mathrm{d}\theta = V_{\text{hi}},$$

as given in Equation (20c). Thus, $\mathcal{J}_{\text{hi}} = c_{\text{hi}} V_{\text{hi}}$.

For the left-hand side of the performance inequality, we take each expectation in $\mathcal{J}_{\text{mf}}$ in turn. We first note that the expected iteration cost of Algorithm 2, $\mathbf{E}(C_{\text{mf}})$, is the sum of the expected cost of a single low-fidelity simulation, and the expected cost of M high-fidelity simulations. By definition, the expected cost of a single low-fidelity simulation $\mathbf{y}_{\text{lo}} \sim f_{\text{lo}}(\cdot \mid \theta)$ across $\theta \sim q(\cdot)$ is given by c_{lo}. Thus the remaining cost, $\mathbf{E}(\delta C_{\text{mf}}) = \mathbf{E}(C_{\text{mf}}) - c_{\text{lo}}$, is the expected cost of M high-fidelity simulations. Conditioning on θ, $\mathbf{y}_{\text{lo}}$ and $M = m$, the expected remaining cost is, by definition,

$$\mathbf{E}(\delta C_{\text{mf}} \mid \theta, \mathbf{y}_{\text{lo}}, M = m) = m c_{\text{hi}}(\theta, \mathbf{y}_{\text{lo}}).$$

Taking expectations over the conditional distribution $M \sim \text{Poi}(\mu(\theta, \mathbf{y}_{\text{lo}}))$, we have

$$\mathbf{E}(\delta C_{\text{mf}} \mid \theta, \mathbf{y}_{\text{lo}}) = \mu(\theta, \mathbf{y}_{\text{lo}}) c_{\text{hi}}(\theta, \mathbf{y}_{\text{lo}}).$$

Finally, integrating this expression over the density ρ in Equation (20f) gives the first factor of Equation (9).

It remains to show that

$$\mathbf{E}(w_{\text{mf}}^2 \Delta^2) = V_{\text{mf}} + \iint \Delta_q(\theta)^2 \frac{\eta(\theta, \mathbf{y}_{\text{lo}})}{\mu(\theta, \mathbf{y}_{\text{lo}})} \rho(\theta, \mathbf{y}_{\text{lo}}) \mathrm{d}\theta \mathrm{d}\mathbf{y}_{\text{lo}}.$$

We first condition on θ, $\mathbf{y}_{\text{lo}}$ and $M = m$, to write

$$\begin{aligned}\mathbf{E}(w_{\text{mf}}^2\Delta^2 \mid \theta, \mathbf{y}_{\text{lo}}, m) &= \Delta_q^2 \mathbf{E}(\omega_{\text{mf}}^2 \mid \theta, \mathbf{y}_{\text{lo}}, m) \\ &= \Delta_q^2 \left[\omega_{\text{lo}}^2 + \frac{2}{\mu}\omega_{\text{lo}}\mathbf{E}\left(D_m \mid \theta, \mathbf{y}_{\text{lo}}\right) + \frac{1}{\mu^2}\mathbf{E}\left(D_m^2 \mid \theta, \mathbf{y}_{\text{lo}}\right)\right],\end{aligned}$$

for the random variable $D_m = \sum_{i=1}^{m}\left(\omega_{\text{hi},i} - \omega_{\text{lo}}\right)$. It is straightforward to show that

$$\begin{aligned}\mathbf{E}(D_m \mid \theta, \mathbf{y}_{\text{lo}}) &= m\mathbf{E}(\omega_{\text{hi}} \mid \theta, \mathbf{y}_{\text{lo}}) - m\omega_{\text{lo}}, \\ \mathbf{E}(D_m^2 \mid \theta, \mathbf{y}_{\text{lo}}) &= m\mathbf{E}\left((\omega_{\text{hi}} - \omega_{\text{lo}})^2 \mid \theta, \mathbf{y}_{\text{lo}}\right) + (m^2 - m)\mathbf{E}(\omega_{\text{hi}} - \omega_{\text{lo}} \mid \theta, \mathbf{y}_{\text{lo}})^2,\end{aligned}$$

where we exploit the conditional independence of the high-fidelity simulations $\mathbf{y}_{\text{hi},i}$ and $\mathbf{y}_{\text{hi},j}$, for $i \neq j$. On substitution of these conditional expectations, we then rearrange to write

$$\mathbf{E}(w_{\text{mf}}^2\Delta^2 \mid \theta, \mathbf{y}_{\text{lo}}, m) = \Delta_q^2\left[\left(1 - \frac{2m}{\mu}\right)\omega_{\text{lo}}^2 + \frac{2m}{\mu}\omega_{\text{lo}}\lambda_{\text{hi}} + \frac{m}{\mu^2}\text{Var}(\omega_{\text{hi}} - \omega_{\text{lo}} \mid \theta, \mathbf{y}_{\text{lo}}) + \left(\frac{m(\lambda_{\text{hi}} - \omega_{\text{lo}})}{\mu}\right)^2\right],$$

where we write the conditional expectation $\lambda_{\text{hi}}(\theta, \mathbf{y}_{\text{lo}}) = \mathbf{E}(\omega_{\text{hi}} \mid \theta, \mathbf{y}_{\text{lo}})$. At this point we can take expectations over M and rearrange to give

$$\begin{aligned}\mathbf{E}(w_{\text{mf}}^2\Delta^2 \mid \theta, \mathbf{y}_{\text{lo}}) &= \Delta_q^2\left[2\omega_{\text{lo}}\lambda_{\text{hi}} - \omega_{\text{lo}}^2 + \frac{1}{\mu}\text{Var}(\omega_{\text{hi}} - \omega_{\text{lo}} \mid \theta, \mathbf{y}_{\text{lo}}) + \frac{(\text{Var}(M \mid \theta, \mathbf{y}_{\text{lo}}) + \mu^2)(\lambda_{\text{hi}} - \omega_{\text{lo}})^2}{\mu^2}\right] \\ &= \Delta_q^2\left[\lambda_{\text{hi}}^2 + \frac{1}{\mu}\left(\text{Var}(\omega_{\text{hi}} - \omega_{\text{lo}} \mid \theta, \mathbf{y}_{\text{lo}}) + \frac{\text{Var}(M \mid \theta, \mathbf{y}_{\text{lo}})}{\mu}\left(\mathbf{E}(\omega_{\text{hi}} - \omega_{\text{lo}} \mid \theta, \mathbf{y}_{\text{lo}})\right)^2\right)\right]. \qquad (22)\end{aligned}$$

Here, we can use the assumption that M conditioned on θ and $\mathbf{y}_{\text{lo}}$ is Poisson distributed, noting that the statement of Theorem 4 can be adapted for other conditional distributions of M with different conditional variance functions. Under the Poisson assumption, we can substitute $\text{Var}(M \mid \theta, \mathbf{y}_{\text{lo}}) = \mu(\theta, \mathbf{y}_{\text{lo}})$ to give

$$\mathbf{E}(w_{\text{mf}}^2\Delta^2 \mid \theta, \mathbf{y}_{\text{lo}}) = \Delta_q^2\left[\lambda_{\text{hi}}^2 + \frac{\mathbf{E}\left(\left(\omega_{\text{hi}} - \omega_{\text{lo}}\right)^2 \mid \theta, \mathbf{y}_{\text{lo}}\right)}{\mu(\theta, \mathbf{y}_{\text{lo}})}\right].$$

Finally, we take expectations with respect to the probability density ρ in Equation (20f), and the product in Equation (9) follows. □

Given the main result from Theorem 4 on the conditions for improvements using multifidelity importance sampling, the following results (Lemma 5 and Corollary 6) demonstrate the form and existence of the mean function, $\mu(\theta, \mathbf{y}_{\text{lo}})$, such that the conditions are satisfied.

Lemma 5. *The functional $\mathcal{J}_{\text{mf}}[\mu]$ quantifying the performance of Algorithm 2 is optimised by the function $\mu^\star$, where*

$$\mu^\star(\theta, \mathbf{y}_{\text{lo}})^2 = \Delta_q(\theta)^2\left[\frac{\eta(\theta, \mathbf{y}_{\text{lo}})/V_{\text{mf}}}{c_{\text{hi}}(\theta, \mathbf{y}_{\text{lo}})/c_{\text{lo}}}\right]. \qquad (23)$$

Proof. We write the functional $\mathcal{J}_{\text{mf}}[\mu] = C[\mu]\mathcal{V}[\mu]$ in Equation (9) as the product of functionals

$$C[\mu] = c_{\text{lo}} + \iint \mu c_{\text{hi}}\ \rho \text{d}\theta\text{d}\mathbf{y}_{\text{lo}}, \qquad (24a)$$

$$\mathcal{V}[\mu] = V_{\text{mf}} + \iint \Delta_q^2\frac{\eta}{\mu}\ \rho\text{d}\theta\text{d}\mathbf{y}_{\text{lo}}. \qquad (24b)$$

Standard 'product rule' results from calculus of variations allow us to write the functional derivative of $\mathcal{J}_{\text{mf}}$ with respect to μ as

$$\begin{aligned}\frac{\delta\mathcal{J}_{\text{mf}}}{\delta\mu} &= \mathcal{V}[\mu]\frac{\delta C}{\delta\mu} + C[\mu]\frac{\delta\mathcal{V}}{\delta\mu} \\ &= \mathcal{V}[\mu]c_{\text{hi}}\rho - C[\mu]\frac{\Delta_q^2\eta\rho}{\mu^2}.\end{aligned}$$

Setting this functional derivative to zero, the optimal function, $\mu^\star$, satisfies

$$\mu^\star(\theta, \mathbf{y}_{\text{lo}})^2 = \frac{C[\mu^\star]}{\mathcal{V}[\mu^\star]}\frac{\Delta_q(\theta)^2\eta(\theta, \mathbf{y}_{\text{lo}})}{c_{\text{hi}}(\theta, \mathbf{y}_{\text{lo}})}. \qquad (25)$$

The result in Equation (10) follows on showing that $C[\mu^\star]/V[\mu^\star] = c_{\text{lo}}/V_{\text{mf}}$.

On substituting Equation (25) into Equation (24) we find

$$C[\mu^\star] = c_{\text{lo}} + \sqrt{\frac{C[\mu^\star]}{\mathcal{V}[\mu^\star]}} \iint \sqrt{\Delta_q^2 \eta c_{\text{hi}}}\ \rho \mathrm{d}\theta \mathrm{d}\mathbf{y}_{\text{lo}},$$

$$\mathcal{V}[\mu^\star] = V_{\text{mf}} + \sqrt{\frac{\mathcal{V}[\mu^\star]}{C[\mu^\star]}} \iint \sqrt{\Delta_q^2 \eta c_{\text{hi}}}\ \rho \mathrm{d}\theta \mathrm{d}\mathbf{y}_{\text{lo}},$$

from which it follows that

$$\sqrt{\frac{\mathcal{V}[\mu^\star]}{C[\mu^\star]}} c_{\text{lo}} = \sqrt{\frac{C[\mu^\star]}{\mathcal{V}[\mu^\star]}} V_{\text{mf}} = \sqrt{C[\mu^\star]\mathcal{V}[\mu^\star]} - \iint \sqrt{\Delta_q^2 \eta c_{\text{hi}}}\ \rho \mathrm{d}\theta \mathrm{d}\mathbf{y}_{\text{lo}}.$$

Multiplying this equation by $\sqrt{C[\mu^\star]\mathcal{V}[\mu^\star]}$, we have $\mathcal{V}[\mu^\star]c_{\text{lo}} = C[\mu^\star]V_{\text{mf}}$, and thus Equation (10) follows from Equation (25). □

Corollary 6. *There exists a mean function, μ, such that the performance of Algorithm 2 exceeds the performance of Algorithm 1, if and only if*

$$\sqrt{\frac{c_{\text{lo}}}{c_{\text{hi}}}\frac{V_{\text{mf}}}{V_{\text{hi}}}} + \iint \sqrt{\frac{\Delta_q(\theta)^2 \eta(\theta, \mathbf{y}_{\text{lo}})}{V_{\text{hi}}}} \sqrt{\frac{c_{\text{hi}}(\theta, \mathbf{y}_{\text{lo}})}{c_{\text{hi}}}}\ \rho(\theta, \mathbf{y}_{\text{lo}}) \mathrm{d}\theta \mathrm{d}\mathbf{y}_{\text{lo}} < 1. \tag{26}$$

Proof. On substituting Equation (23) into Equation (24), we find that the condition $\mathcal{J}_{\text{mf}}^\star = \mathcal{J}_{\text{mf}}[\mu^\star] < \mathcal{J}_{\text{hi}} = c_{\text{hi}} V_{\text{hi}}$ is equivalent to

$$\left(\sqrt{c_{\text{lo}} V_{\text{mf}}} + \iint \sqrt{\Delta_q(\theta)^2 \eta(\theta, \mathbf{y}_{\text{lo}}) c_{\text{hi}}(\theta, \mathbf{y}_{\text{lo}})}\ \rho(\theta, \mathbf{y}_{\text{lo}}) \mathrm{d}\theta \mathrm{d}\mathbf{y}_{\text{lo}} \right)^2 < c_{\text{hi}} V_{\text{hi}}.$$

A simple rearrangement of this inequality gives the inequality in Equation (26). □

To interpret the condition in Equation (26), we note that the first term is determined by (a) our assumption of a significant reduction in simulation burden of the low-fidelity model over the high-fidelity model, $c_{\text{lo}} < c_{\text{hi}}$, and (b) the ratio of the two integrals,

$$\frac{V_{\text{mf}}}{V_{\text{hi}}} = \frac{\int \Delta_q(\theta)^2 \mathbf{E}(\mathbf{E}(\omega_{\text{hi}} \mid \theta, \mathbf{y}_{\text{lo}})^2 \mid \theta) q(\theta)\ \mathrm{d}\theta}{\int \Delta_q(\theta)^2 \mathbf{E}(\omega_{\text{hi}}^2 \mid \theta) q(\theta)\ \mathrm{d}\theta}.$$

Exploiting the law of total variance, we note that

$$\begin{aligned} \mathbf{E}(\omega_{\text{hi}}^2 \mid \theta) &= \mathrm{Var}(\omega_{\text{hi}} \mid \theta) + L_{\omega_{\text{hi}}}(\theta)^2, \\ \mathbf{E}(\mathbf{E}(\omega_{\text{hi}} \mid \theta, \mathbf{y}_{\text{lo}})^2 \mid \theta) &= \mathrm{Var}(\mathbf{E}(\omega_{\text{hi}} \mid \theta, \mathbf{y}_{\text{lo}}) \mid \theta) + L_{\omega_{\text{hi}}}(\theta)^2 \\ &= \mathbf{E}(\omega_{\text{hi}}^2 \mid \theta) - \mathbf{E}(\mathrm{Var}(\omega_{\text{hi}} \mid \theta, \mathbf{y}_{\text{lo}}) \mid \theta). \end{aligned}$$

These equalities imply that

$$\mathbf{E}(\omega_{\text{hi}} \mid \theta)^2 \leq \mathbf{E}(\mathbf{E}(\omega_{\text{hi}} \mid \theta, \mathbf{y}_{\text{lo}})^2 \mid \theta) \leq \mathbf{E}(\omega_{\text{hi}}^2 \mid \theta),$$

where the lower bound is achieved for $\mathbf{y}_{\text{hi}}$ independent of $\mathbf{y}_{\text{lo}}$, while the upper bound would be achieved if $\mathbf{y}_{\text{hi}}$ were a deterministic function of $\mathbf{y}_{\text{lo}}$. In particular, $V_{\text{mf}}/V_{\text{hi}} \leq 1$, and so the first term of Equation (26) is small whenever the low-fidelity model provides significant computational savings versus the high-fidelity model.

The second term in Equation (26) quantifies the detriment to the performance of Algorithm 2 that arises from the inaccuracy of ω_{lo} as an estimate of ω_{hi}. The function $\eta(\theta, \mathbf{y}_{\text{lo}}) = \mathbf{E}((\omega_{\text{hi}} - \omega_{\text{lo}})^2 \mid \theta, \mathbf{y}_{\text{lo}})$ is integrated across the density ρ, weighted by the relative computational cost of the high-fidelity simulation, $c_{\text{hi}}(\theta, \mathbf{y}_{\text{lo}})/c_{\text{hi}}$, and by the contribution of $G(\theta)$ to the variance of the estimated posterior expectation of G. We can conclude that the multifidelity approach requires that the low-fidelity model is accurate in the regions of parameter space where high-fidelity simulations are particularly expensive.

To summarise: if (a) the ratio between average low-fidelity simulation costs and high-fidelity simulation costs is suitably small, and (b) the average disagreement between likelihood-free weightings, as measured by η, is suitably small, then Equation (26) will be satisfied and thus a mean function, $\mu^\star$, exists such that Algorithm 2 is more efficient than Algorithm 1. The optimisation goal of the adaptive scheme proposed in Algorithm 3 optimally tunes the mean function $\mu(\theta, \mathbf{y}_{\text{lo}})$ to maximise the computational benefit.

5. Example: biochemical reaction network

The following example considers the stochastic simulation of a biochemical reaction motif. Readers unfamiliar with these techniques are referred to detailed expositions of by Schnoerr et al. [32], Warne et al. [33] and Erban and Chapman [34]. We model the conversion (over time $t \geq 0$) of substrate molecules, labelled S, into molecules of a product, P. The conversion of S into P

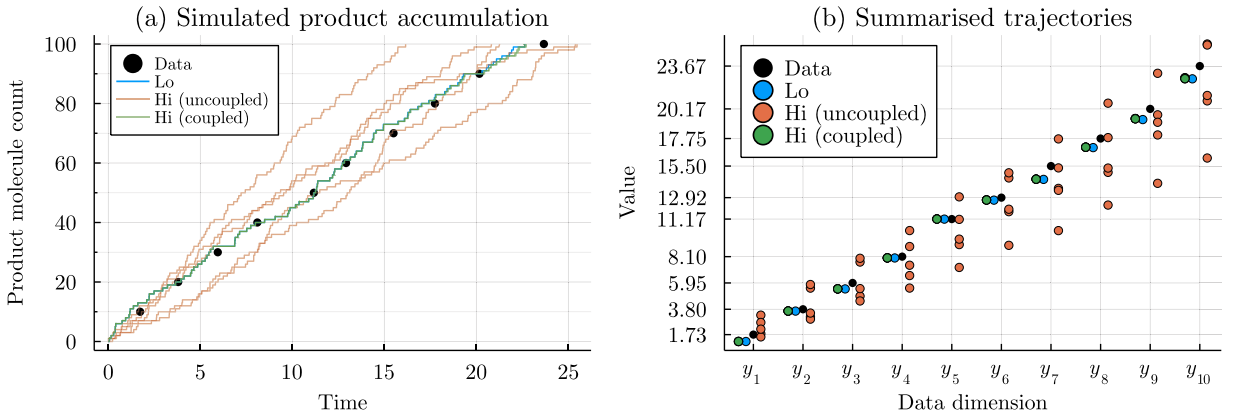

Fig. 1. Effect of multifidelity coupling. (a) Example stochastic trajectories from the high and low-fidelity enzyme kinetics models in Equations (27) and (30) for parameters $\theta = (k_1, k_2, k_3) = (50, 50, 1)$, compared with data used for inference. For one low-fidelity simulation, we generate five uncoupled simulations and five coupled simulations. (b) Ten-dimensional data summarising simulated trajectories in (a). Black represents observed data, y_0; the single low-fidelity simulation $y_{\text{lo}} \sim f_{\text{lo}}(\cdot \mid \theta)$ is in blue; five uncoupled simulations $y_{\text{hi}} \sim f_{\text{hi}}(\cdot \mid \theta)$ are in orange; five coupled simulations $y_{\text{hi}} \sim f_{\text{hi}}(\cdot \mid \theta, y_{\text{lo}})$ are in green (almost coincident). (For interpretation of the colours in the figure(s), the reader is referred to the web version of this article.)

is catalysed by the presence of enzyme molecules, E, which bind with S to form a substrate-enzyme complex, labelled C. After non-dimensionalising units of time and volume, this network motif is represented by three reactions,

$$\mathrm{S} + \mathrm{E} \underset{k_2}{\overset{k_1}{\rightleftharpoons}} \mathrm{C} \xrightarrow{k_3} \mathrm{P} + \mathrm{E}, \tag{27a}$$

parameterised by the vector $\theta = (k_1, k_2, k_3)$ of positive parameters, k_1, k_2, and k_3, and three propensity functions,

$$v_1(t) = k_1 S(t) E(t), \tag{27b}$$

$$v_2(t) = k_2 C(t), \tag{27c}$$

$$v_3(t) = k_3 C(t), \tag{27d}$$

where the integer-valued variables $S(t)$, $E(t)$, $C(t)$ and $P(t)$ represent the molecule numbers at time $t > 0$. The stoichiometric matrix for this model is

$$h = \begin{bmatrix} -1 & 1 & 0 \\ -1 & 1 & 0 \\ 1 & -1 & -1 \\ 0 & 0 & 1 \end{bmatrix}. \tag{28}$$

At $t = 0$, we assume there are no complex or product molecules, but set positive integer numbers $S_0 = 100$ and $E_0 = 5$ of substrate and enzyme molecules, respectively. Given the fixed initial conditions, the parameters in θ are sufficient to specify the dynamics of the model in Equation (27a). The model is stochastic, and induces a distribution, which we denote $f(\cdot \mid \theta)$, on the space of trajectories $x : t \mapsto (S(t), E(t), C(t), P(t))$ of molecule numbers in $\mathbb{N}^4$ over time. Such trajectories evolve according to the discrete-state Markov process

$$x_t = x_0 + \sum_{j=1}^{3} \mathcal{P}_j \left(\int_0^t v_j(s)\, \mathrm{d}s \right) h_{*j}, \tag{29}$$

where $\mathcal{P}_1(\cdot)$, $\mathcal{P}_3(\cdot)$, and $\mathcal{P}_3(\cdot)$, are inhomogeneous Poisson processes.

For the purposes of this example, the observed data (Fig. 1) is given by

$$y_0 = (y_1, \ldots, y_{10}) = (1.73, 3.80, 5.95, 8.10, 11.17, 12.92, 15.50, 17.75, 20.17, 23.67),$$

where the n-th observation represents the hitting time for the product molecule levels exceeding $10n$, that is, $y_n = t$ such that $P(t) = 10n$. We set a prior $\pi(\theta)$ on the vector θ, equal to a product of independent uniform distributions such that $k_1, k_2 \sim \mathrm{U}(10, 100)$ and $k_3 \sim \mathrm{U}(0.1, 10)$. We seek the posterior distribution $\pi(\theta \mid y_0)$ using the likelihood, denoted $\mathcal{L}(\theta) = f(y_0 \mid \theta)$, focusing on the posterior expectation of the function $G(\theta) = k_3$, denoting the rate of conversion of substrate–enzyme complex to product. All code for this example is available at `github.com/tpprescott/mf-lf`, using stochastic simulations implemented by `github.com/tpprescott/ReactionNetworks.jl`. We assume that we cannot calculate the likelihood function, $\mathcal{L}(\theta) = f(y_0 \mid \theta)$, and therefore resort to our likelihood-free framework.

5.1. Multifidelity approximate Bayesian computation

Here we formulate the weighting function to implement ABC importance sampling to compare with an equivalent ABC implementation of adaptive multifidelity importance sampling. See Appendix A for the general formulation of ABC within our framework.

5.1.1. ABC importance sampling

Given θ, the model in Equation (27) can be exactly simulated using the Gillespie stochastic simulation algorithm, to produce draws $y \sim f(\cdot \mid \theta)$ from the exact model [26,34,35]. We will use the ABC likelihood-free weighting with threshold value $\epsilon = 5$ on the Euclidean distance of the simulation from y_0, such that

$$\omega(\theta, y) = \mathbf{1}(\|y - y_0\|_2 < 5),$$

to define the likelihood-free approximation to the posterior, $L_{\text{ABC}}(\theta) = \mathbf{E}(\omega \mid \theta)$. For simplicity of demonstration we set the importance distribution to be equal to the prior, that is, $q = \pi$. In this instance, the likelihood-free weighting in Algorithm 1 reduces to a rejection sampling approach, setting the importance distribution equal to the prior. Furthermore, configuring Algorithm 1 and Algorithm 3 to use the same proposal ensures any performance improvements are due to the multifidelity scheme rather than tuning of proposals.

5.1.2. Multifidelity ABC

The exact Gillespie stochastic simulation algorithm can incur significant computational burden. In the specific case of the network in Equation (27), if the reaction rates k_1 and k_2 are large relative to k_3, there are large numbers of binding/unbinding reactions $\text{S} + \text{E} \longleftrightarrow \text{C}$ that occur in any simulation. In comparison, the reaction $\text{C} \longrightarrow \text{P} + \text{E}$ can only fire exactly 100 times. Michaelis–Menten dynamics exploit this scale separation to approximate the enzyme kinetics network motif. We approximate the conversion of substrate into product as a single reaction step,

$$\text{S} \xrightarrow{k_{\text{MM}}(t)} \text{P}, \tag{30a}$$

where the time-varying rate of conversion, $k_{\text{MM}}(t)$, given by

$$k_{\text{MM}}(t) = \frac{k_3 \min(S(t), E_0)}{K_{\text{MM}} + S(t)}, \tag{30b}$$

$$K_{\text{MM}} = \frac{k_2 + k_3}{k_1}, \tag{30c}$$

induces the propensity function $\upsilon_{\text{MM}}(t) = k_{\text{MM}}(t)S(t)$. We assume initial conditions of $S(0) = S_0 = 100$ and $P(0) = 0$, and fix the parameter $E_0 = 5$. Thus, the parameter vector, $\theta = (k_1, k_2, k_3)$, again fully determines the dynamics of the low-fidelity model in Equation (30). We write $y_{\text{lo}} \sim f_{\text{lo}}(\cdot \mid \theta)$ as the conditional probability density for the Gillespie simulation of the approximated model in Equation (30), where y_{lo} is the vector of ten simulated time points $y_{\text{lo},n}$ at which $10n$ product molecules have been produced.

For a biochemical reaction network consisting of R reactions, the Gillespie simulation algorithm is a deterministic transformation of R independent unit-rate Poisson processes, one for each reaction channel. We can couple the models in Equations (27) and (30) by using the same Poisson process for the single reaction in Equation (30) and for the product formation $\text{C} \longrightarrow \text{P} + \text{E}$ reaction of Equation (27) [11,36]. Using this coupling approach, we first simulate $y_{\text{lo}} \sim f_{\text{lo}}(\cdot \mid \theta)$ from Equation (30). We then produce the coupled simulation $y_{\text{hi}} \sim f_{\text{hi}}(\cdot \mid \theta, y_{\text{lo}})$ from the model in Equation (27), using the shared Poisson process. We set the corresponding likelihood-free weightings to

$$\omega_{\text{hi}}(\theta, y_{\text{hi}}) = \mathbf{I}(|y_{\text{hi}} - y_0| < 5),$$

$$\omega_{\text{lo}}(\theta, y_{\text{lo}}) = \mathbf{I}(|y_{\text{lo}} - y_0| < 5),$$

noting that $\mathbf{E}(\omega_{\text{hi}} \mid \theta) = L_{\text{ABC}}(\theta)$ is the high-fidelity ABC approximation to the likelihood. Fig. 1 illustrates the effect of coupling between low-fidelity and high-fidelity models. The five coupled high-fidelity simulations are significantly less variable than the independent high-fidelity simulations, appearing almost coincident in Fig. 1. This ensures a large degree of correlation between the coupled likelihood-free weightings, ω_{hi} and ω_{lo}. Thus, coupling ensures that ω_{lo} is a reliable proxy for ω_{hi} for use in multifidelity likelihood-free inference.

We implement Algorithm 3 by setting a burn-in period of $N_0 = 10,000$, for which we generate $m_i \sim \text{Poi}(1)$ high-fidelity simulations at each iteration, $i \leq N_0$. Once the burn-in period is complete, we define the partition $\mathcal{D}$ by learning a decision tree through a simple regression, as described in Section 3.4. For iterations $i > N_0$ beyond the burn-in period, we set a step size of $\delta = 10^3$ for the gradient descent update in Equation (14).

5.1.3. Results

Algorithm 1 was run four times, setting the total number of weighted samples to $N = 10,000$, $N = 20,000$, $N = 40,000$ and $N = 80,000$. Similarly, Algorithm 3 was run five times, setting $N = 40,000$, $N = 80,000$, $N = 160,000$, $N = 320,000$ and $N = 640,000$.

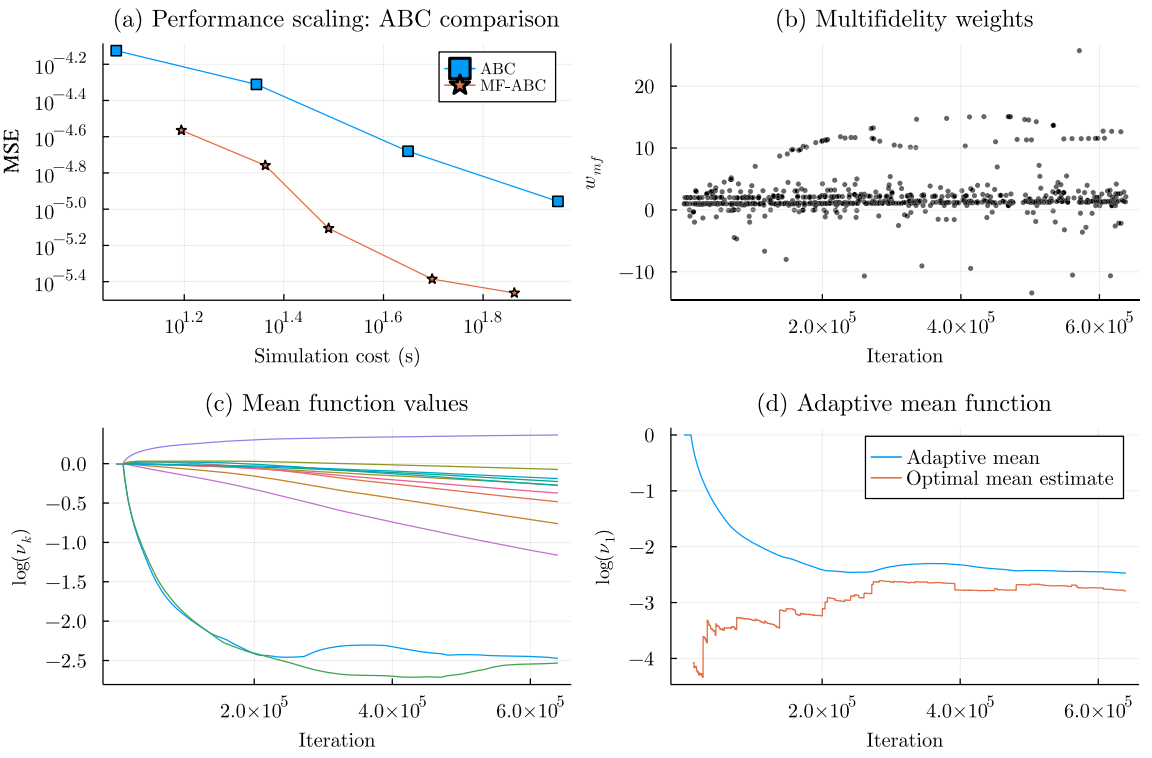

Fig. 2. Multifidelity ABC. (a) Total simulation cost versus the empirical mean-squared error (MSE) of output estimate, $\hat{G}$, for four runs of Algorithm 1 (ABC) and five runs of Algorithm 3 (MF-ABC). (b) Values of the multifidelity weight, w_{mf}, during the longest run of Algorithm 3, for iterations where $\omega_{\text{mf}} \neq \omega_{\text{lo}}$, such that the low-fidelity likelihood-free weighting is corrected based on at least one high-fidelity simulation. (c) The evolution of the values of $v_k^{(i)}$ during the longest run of Algorithm 3, here different coloured lines are used to distinguish $v_k^{(i)}$ between distinct partitions, D_k, however, the colour itself has no specific interpretation. (d) A comparison of the adaptive $v_1^{(i)}$ to the evolving best estimate of the optimal $v_1^\star$, given by Equation (13), based on the Monte Carlo estimates in Equation (19).

Since we do not have access to the true posterior mean, $\mathbf{E}_{\pi_{\omega_{\text{hi}}}}(G|y_0)$, we use the empirical mean over all high-fidelity runs as a proxy, resulting in an empirical MSE estimate for the multifidelity scheme. Fig. 2a shows how the empirical MSE in the estimate, $\hat{G}$, varies with the total simulation cost, C_{tot}, shown for each of the two algorithms. The slope of each curve (on a log-log scale) is approximately -1, corresponding to the dominant behaviour of the MSE being reciprocal with total simulation time, as observed in Equation (7). The offset in the two curves corresponds to the inequality $\mathcal{J}_{\text{mf}} < \mathcal{J}_{\text{hi}}$ in the leading order coefficient, thereby demonstrating the improved performance of Algorithm 3 over Algorithm 1.

The values in Fig. 2b show the multifidelity weights, w_i. We show only those weights not equal to zero or one, corresponding to those iterations where $\omega_{\text{lo}}(\theta_i, y_{\text{lo},i})$ has been corrected by at least one $\omega_{\text{hi}}(\theta_i, y_{\text{hi},i,j}) \neq \omega_{\text{lo}}(\theta_i, y_{\text{lo},i})$. Clearly there is a significant amount of correction applied to the low-fidelity weights. However, as demonstrated by the improved performance statistics, Algorithm 3 has learned the required allocation of computational budget to the high-fidelity simulations that balances the trade-off between achieving reduced overall simulation times and correcting inaccuracies in the low-fidelity simulation.

Each run of Algorithm 3 includes a burn-in period of $10,000$ iterations, at the conclusion of which a partition $\mathcal{D}$ is created, based on decision tree regression. In Appendix C, we show how this decision tree is used to define a piecewise-constant mean function, specifically for the partition $\mathcal{D}$ used for the final run of Algorithm 3 (i.e. $N = 640,000$ iterations). In Fig. 2c, we show the evolution of the values of $v_k^{(i)}$ used in this mean function, over iterations i. Following the updating rule in Equation (19), the trajectory of $v_k^{(i)}$ converges exponentially towards a Monte Carlo estimate of the optimal value $v_k^\star$ given in Equation (13). However, we can see from Fig. 2c that, as more simulations are completed and the Monte Carlo estimates in Equation (19) evolve, the values of each parameter, v_k, track updated estimates. This is illustrated in Fig. 2d for v_1, where the estimated optimum $v_1^\star$ evolves as more simulations are completed. We note that the gradient descent update in Equation (19) at iteration i depends on *all* $v_k^{(i)}$ values. Thus, the observed convergence of $v_1^{(i)}$ to the evolving estimate of $v_1^\star$ is not necessarily monotonic.

Fig. 2d illustrates the motivation for the use of gradient descent rather than simply using the analytically obtained optimum. When very few simulations have been completed, then the estimates in Equation (10) are small and their ratios are numerically unstable, and often far from the true optimum. If $v_k^{(i)}$ values are too small in early iterations, then estimates become *more* numerically unstable, since fewer high-fidelity simulations are completed for small μ. Instead, using gradient descent ensures that enough high-fidelity simulations are completed for each $\mathcal{D}_k$, including those with low volume under the measure ρ, to stabilise the estimates required in Equation (10) and thus stabilise the multifidelity algorithm.

5.2. Multifidelity Bayesian synthetic likelihood

Consider the same model of enzyme kinetics as in Section 5.1. As depicted in Fig. 1, this model has low-fidelity (Michaelis–Menten) stochastic dynamics with distribution $f_{\text{lo}}(\cdot \mid \theta)$, and coupled high-fidelity stochastic dynamics with distribution $f_{\text{hi}}(\cdot \mid \theta, y_{\text{lo}})$. We now redefine ω_{lo} and ω_{hi} to be Bayesian synthetic likelihoods, based on K pairs of coupled simulations,

$$y_{\text{lo},k} \sim f_{\text{lo}}(\cdot \mid \theta),$$
$$y_{\text{hi},k} \sim f_{\text{hi}}(\cdot \mid \theta, y_{\text{lo},k}),$$

for $k = 1, \ldots, K$. That is,

$$\omega_{\text{lo}}(\theta, \mathbf{y}_{\text{lo}}) = \mathcal{N}\left(y_0 : \mu(\mathbf{y}_{\text{lo}}), \Sigma(\mathbf{y}_{\text{lo}})\right),$$
$$\omega_{\text{hi}}(\theta, \mathbf{y}_{\text{hi}}) = \mathcal{N}\left(y_0 : \mu(\mathbf{y}_{\text{hi}}), \Sigma(\mathbf{y}_{\text{hi}})\right),$$

are the Gaussian likelihoods of the observed data, under the empirical mean and covariance of K low-fidelity and (coupled) high-fidelity simulations, respectively. Just like in the ABC example, for simplicity of demonstration we set the importance distribution to be equal to the prior, that is, $q = \pi$.

Algorithm 1 was run three times, using $\omega_{\text{hi}}(\theta, \mathbf{y}_{\text{hi}})$ dependent on high-fidelity simulations $\mathbf{y}_{\text{hi}} \sim f(\cdot \mid \theta)$, alone, and setting the number of iterations to $N = 2,500$, $N = 5,000$ and $N = 10,000$. Similarly, Algorithm 3 was run four times using the coupled multifidelity model, setting the number of iterations to $N = 4,000$, $N = 8,000$, $N = 16,000$ and $N = 32,000$, and initialising with a burn-in of size $N_0 = 2,000$. The adaptive step size is set to $\delta = 10^8$. In both algorithms, we set the number of simulations required for each evaluation of $\omega_{\text{hi}}(\theta, (y_{\text{hi},1}, \ldots, y_{\text{hi},K}))$ or $\omega_{\text{lo}}(\theta, (y_{\text{lo},1}, \ldots, y_{\text{lo},K}))$ as $K = 100$.

Fig. 3 depicts the performance of multifidelity BSL inference, where Algorithm 3 is applied with BSL likelihood-free weightings, ω_{lo} and ω_{hi}. As with MF-ABC, Fig. 3a shows that the MF-BSL generates improved performance over high-fidelity BSL inference, achieving lower MSE estimates for a given computational budget. We also note in Fig. 3a that the curve corresponding to MF-BSL has slope less than -1. This is due to (a) the overhead cost of the initial burn-in period of Algorithm 3, and also (b) the conservative convergence of $v^{(i)}$ to the optimum, as shown in Fig. 3c–d. Both observations imply that earlier iterations are less efficiently produced than later iterations, meaning that larger samples show greater improvements than expected from the reciprocal relationship in Equation (7).

Comparing Fig. 3b to Fig. 2b, we note that there are very few negative multifidelity weightings in MF-BSL, in comparison to MF-ABC. We can conclude that the Bayesian synthetic likelihood, constructed using low-fidelity simulations, tends to underestimate the likelihood of the observed data compared to using high-fidelity simulations. We note also in this comparison that the multifidelity likelihood-free weightings are on significantly different scales.

6. Discussion

The characteristic computational burden of simulation-based, likelihood-free Bayesian inference methods is often a barrier to their successful implementation. Multifidelity simulation techniques have previously been shown to improve the efficiency of likelihood-free inference in the context of ABC. In this work, we have demonstrated that these techniques can be readily applied to a range of likelihood-free approaches. Furthermore, we have introduced a computational methodology for automating the multifidelity approach, adaptively allocating simulation resources across different fidelities in order to ensure near-optimal efficiency gains from this technique. As parameter space is explored, our methodology, given in Algorithm 3, learns the relationships between simulation accuracy and simulation costs at the different fidelities, and adapts the requirement for high-fidelity simulation accordingly.

The multifidelity approach to likelihood-free inference is one of a number of strategies for speeding up inference, which include MCMC and SMC sampling techniques [7–9] and methods for variance reduction such as multilevel estimation [16,18,17,29]. A key observation in the previous work of Prescott and Baker [12] and Warne et al. [15] is that applying multifidelity techniques provides 'orthogonal' improvements that combine synergistically with these other established approaches to improving efficiency. Similarly, we envision that Algorithm 3 can be adapted into an SMC or multilevel algorithm with minimal difficulty, following the templates set by Prescott and Baker [12] and Warne et al. [15].

The multifidelity approach discussed in this work is a highly flexible generalisation of existing multifidelity techniques, which can be viewed as special cases of Algorithm 2. In each of MF-ABC [11,12], LZ-ABC [20], and DA-ABC [22], it is assumed that ω_{hi} is an ABC likelihood-free weighting, which we relax in this work. Furthermore, LZ-ABC and DA-ABC both use $\omega_{\text{lo}} \equiv 0$, so that parameters are always rejected if no high-fidelity simulation is completed. We relax this assumption to allow for any low-fidelity likelihood-free weighting. In all of MF-ABC, LZ-ABC and DA-ABC, the conditional distribution of M, given a parameter value θ and low-fidelity simulation output $\mathbf{y}_{\text{lo}}$ is Bernoulli distributed, with mean $\mu(\theta, \mathbf{y}_{\text{lo}}) \in (0, 1]$. In this work we change this distribution to Poisson, to ease analytical results, but any conditional distribution for M can be used. These adaptations are explored further in Appendix B.

In the case of MF-ABC (as originally formulated by [11]) and DA-ABC [21,22], the mean function, $\mu(\theta, y_{\text{lo}})$, depends on a single low-fidelity simulation and is assumed to be piecewise constant in the value of the indicator function $\mathbf{1}(d(y_{\text{lo}}, y_0) < \epsilon)$. Lazy ABC is more generic in its definition of $\mu = \mu(\phi(\theta, \mathbf{y}_{\text{lo}}))$ to depend on the value of any *decision statistic*, ϕ. In this work, we consider more general piecewise constant mean functions, $\mu_{\mathcal{D}}$, for heuristically derived partitions $\mathcal{D}$ of $(\theta, \mathbf{y}_{\text{lo}})$-space. We observe that $(\theta, \mathbf{y}_{\text{lo}})$ may be of very high dimension; in the BSL example in Section 5.2, having $K = 100$ low-fidelity simulations $y_{\text{lo}} \in \mathbb{R}^{10}$ means that the input

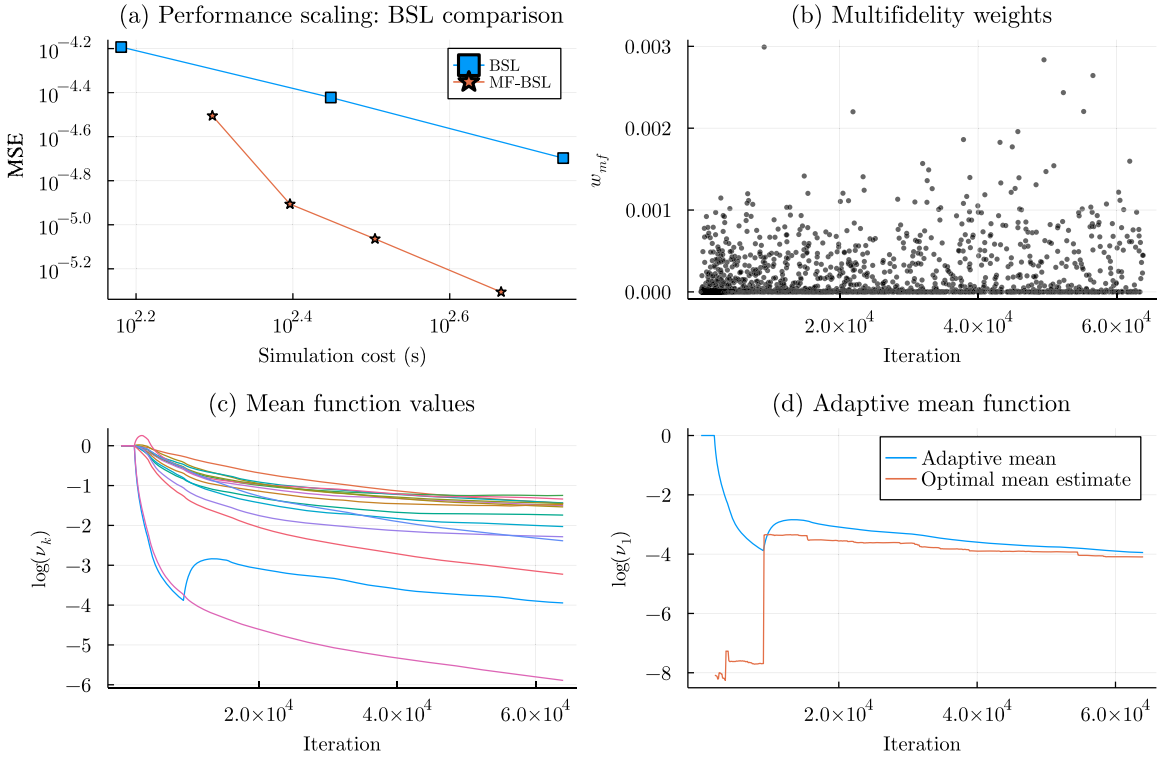

Fig. 3. Multifidelity BSL. (a) Total simulation cost versus the empirical MSE of output estimate, $\hat{G}$, for three runs of Algorithm 1 (BSL) and four runs of Algorithm 3 (MF-BSL). (b) Values of the multifidelity weight, w_{mf}, during the longest run of Algorithm 3, for iterations where $\omega_{\text{mf}} \neq \omega_{\text{lo}}$, such that the low-fidelity likelihood-free weighting is corrected based on at least one high-fidelity simulation. (c) The evolution of the values of $\nu_k^{(i)}$ during the longest run of Algorithm 3, here different coloured lines are used to distinguish $\nu_k^{(i)}$ between distinct partitions, D_k, however, the colour itself has no specific interpretation. (d) A comparison of the adaptive $\nu_1^{(i)}$ to the evolving best estimate of the optimal $\nu_1^\star$, given by Equation (13), based on the Monte Carlo estimates in Equation (19).

to μ is of dimension 1003. In this situation, it may be tempting to seek a mean function that only depends on θ. However, we recall that the optimal mean function, $\mu^\star(\theta, \mathbf{y}_{\text{lo}})$, derived in Lemma 5, depends on the conditional expectation $\mathbf{E}((\omega_{\text{hi}} - \omega_{\text{lo}})^2 \mid \theta, \mathbf{y}_{\text{lo}})$. Thus, by ignoring $\mathbf{y}_{\text{lo}}$, we would ignore the information about ω_{hi} given by the evaluation of $\omega_{\text{lo}}(\theta, \mathbf{y}_{\text{lo}})$. Furthermore, the high dimension of the inputs to $\mu^\star$ suggest that this function is not necessarily well-approximated by a decision tree. Future work may focus on methods to learn the optimal mean function directly without resorting to piecewise constant approximations [37]. The key problem is ensuring the conservatism of any alternative estimate of $\mu^\star$, recalling that the variance of w_{mf} is inversely proportional to μ.

In the example explored in Section 5, we considered the use of Algorithm 3 where ω_{hi} and ω_{lo} were first both ABC likelihood-free weightings, and then both BSL likelihood-free weightings. In principle, this method should also allow for ω_{lo} to be, for example, an ABC likelihood-free weighting based on a single low-fidelity simulation, and ω_{hi} to be a BSL likelihood-free weighting based on $K > 1$ high-fidelity simulations. However, the success of the multifidelity method depends explicitly on the function $\eta(\theta, \mathbf{y}_{\text{lo}}) = \mathbf{E}((\omega_{\text{hi}} - \omega_{\text{lo}})^2 \mid \theta, \mathbf{y}_{\text{lo}})$ being sufficiently small, as quantified in Corollary 6. If ω_{lo} and ω_{hi} are on different scales, as is likely when one is an ABC weighting and one a BSL weighting, then this function is not sufficiently small in general, and so the multifidelity approach fails. We note, however, that we could instead consider the scaled low-fidelity weighting, $\tilde{\omega}_{\text{lo}} = \gamma\omega_{\text{lo}}$, in place of ω_{lo} in Algorithms 2 and 3 with no change to the target distribution. Here, γ is an additional parameter that can be tuned with μ when minimising the performance metric, $\mathcal{J}_{\text{mf}}$; the optimal value of this parameter would need to be learned in parallel with the optimal mean function, μ. We defer this adaptation to future work.

The analysis of performance improvements for multifidelity importance sampling over high-fidelity likelihood-free importance sampling assumes the importance distribution, $q(\theta)$, is the same in both methods. As mentioned in Section 2, this importance distribution can also be tuned to improve efficiency in a similar way to the cost/variance trade-off that we optimise in the multifidelity scheme. While we have not analysed this here, we expect that tuning the proposal distribution could also provide additional improvements. While the analysis may be non-trivial, standard adaptive importance sampling could be applied to tune a proposal from fixed parametric family [38], or more generally, sequential Monte Carlo methods can be used to deal with this problem practically without assuming a parametric form for $q(\theta)$ [12]. Future work should consider the performance improvements available when the assumption of a fixed proposal is relaxed.

Finally, this work follows Prescott and Baker [11,12] in considering only a single low-fidelity model. There is significant scope for further improvements by applying these approaches to suites of low-fidelity approximations [39]. For example, exact stochastic simulations of biochemical networks, such as that simulated in Section 5, may also be approximated by tau-leaping [33,40], where the

time discretisation parameter τ tends to be chosen to trade off computational savings against accuracy: exactly the trade-off explored in this work. Clearly, this parameter therefore has important consequences for the success of a multifidelity inference approach using such an approximation strategy. There are several natural extensions that could be applied to include multiple fidelities (such as multiple τ resolutions). As an example, suppose three levels of fidelity, *low*, *medium* and *high* with respective weights $\omega_{\text{lo}}, \omega_{\text{med}}$ and ω_{hi}. In this setting we can apply the multifidelity weighting (Equation (4)) recursively to obtain

$$\omega_{\text{mf}}(\theta, \mathbf{z}) = \omega_{\text{lo}}(\theta, \mathbf{y}_{\text{lo}}) + \frac{1}{\mu_{\text{med}}(\theta, \mathbf{y}_{\text{lo}})} \sum_{i=1}^{m} \left[\left\{ \omega_{\text{med}}(\theta, \mathbf{y}_{\text{med},i}) + \frac{1}{\mu_{\text{hi}}(\theta, \mathbf{y}_{\text{med},i})} \sum_{j=1}^{n_i} \left[\omega_{\text{hi}}(\theta, \mathbf{y}_{\text{hi},j}) - \omega_{\text{med}}(\theta, \mathbf{y}_{\text{med},i}) \right] \right\} - \omega_{\text{lo}}(\theta, \mathbf{y}_{\text{lo}}) \right],$$

where m is the random number of medium-fidelity simulations conditional on the parameters and the low-fidelity simulation and n_i is the random number of high-fidelity simulations conditional on the parameters and the ith medium-fidelity simulation. The mean functions of these random variables are $\mu_{\text{med}}(\theta, \mathbf{y}_{\text{lo}})$ and $\mu_{\text{hi}}(\theta, \mathbf{y}_{\text{med},i})$, respectively. Alternatively, we could consider a set of J low-fidelity models $\omega_{\text{lo}}^1, \ldots, \omega_{\text{lo}}^J$, that may have different accuracy properties in different partitions of parameter space $D_1, \ldots, D_J$. The multifidelity weighting can be formulated as

$$\omega_{\text{mf}}(\theta, \mathbf{z}) = \sum_{j=1}^{J} \mathbf{I}(\theta \in D_j) \left\{ \omega_{\text{lo}}^j(\theta, \mathbf{y}_{\text{lo}}^j) + \frac{1}{\mu^j(\theta, \mathbf{y}_{\text{lo}}^j)} \sum_{i=1}^{m} \left[\omega_{\text{hi}}(\theta, \mathbf{y}_{\text{hi},i}) - \omega_{\text{lo}}(\theta, \mathbf{y}_{\text{lo}}^j) \right] \right\},$$

assuming that model ω_{lo}^j has the optimal accuracy characteristics in partition D_j. Of course, it may not be known *a priori* which low-fidelity model performs best in a given partition, and therefore we may choose to include the choice of partition as part of the adaptive tuning scheme. In future work, a full exploration of the use of *multiple* low-fidelity model approximations will be vital for the full potential of multifidelity likelihood-free inference to be realised. The theory and practice presented here along with clear paths for potential extension have the potential to achieve substantial performance gains in likelihood-free inference.

CRediT authorship contribution statement

Thomas P. Prescott: Conceptualization, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **David J. Warne:** Conceptualization, Formal analysis, Funding acquisition, Methodology, Validation, Writing – review & editing. **Ruth E. Baker:** Conceptualization, Funding acquisition, Methodology, Supervision, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

All code shared on Github, as detailed in the manuscript.

Acknowledgements

REB and TPP acknowledge funding for this work through the BBSRC/UKRI grant BB/R00816/1. TPP is supported by the Alan Turing Institute and by Wave 1 of the UKRI Strategic Priorities Fund, under the “Shocks and Resilience” theme of the EPSRC/UKRI grant EP/W006022/1. DJW thanks the Australian Mathematical Society for a Lift-off Fellowship and the Queensland University of Technology (QUT) for support through the Early Career Researcher Support Scheme. DJW acknowledges continued support from the Centre for Data Science at QUT. REB is supported by a Royal Society Wolfson Research Merit Award.

Appendix A. Expressions for standard likelihood-free approaches

Here, we briefly highlight how standard likelihood-free approaches can be considered as special cases of our general formulation in Section 2 of the main manuscript.

A.1. Approximate Bayesian computation

Approximate Bayesian computation is a widely-used example of likelihood-free inference, where

$$\omega_{\text{ABC}}(\theta, \mathbf{y}) = \frac{1}{K}\sum_{k=1}^{K} \mathbf{I}(d(y_k, y_0) \leq \epsilon),$$

is the random fraction of K simulations, y_k, which are within a distance of ϵ of the observed data, y_0, as measured by the metric d [4]. In most cases it is standard to set $K = 1$ [8]. Taking the conditional expectation of ω_{ABC}, given θ, this likelihood-free weighting induces the ABC approximation to the likelihood,

$$L_{\text{ABC}}(\theta) = \mathbf{E}(\omega_{\text{ABC}} \mid \theta) = \mathbf{P}(d(y, y_0) \leq \epsilon \mid \theta),$$

for any K. Under appropriate choices of d and ϵ, the approximate likelihood function, $L_{\text{ABC}}(\theta)$, may be considered approximately proportional to the likelihood, $\mathcal{L}(\theta)$.

A.2. Bayesian synthetic likelihood

The BSL approach replaces the true likelihood with a Monte Carlo likelihood-free weighting based on a Gaussian density [6],

$$\omega_{\text{SL}}(\theta, \mathbf{y}) = \mathcal{N}(y_0; \hat{\mu}(\mathbf{y}), \hat{\Sigma}^2(\mathbf{y})),$$

with empirical mean $\hat{\mu}(\mathbf{y}) = \sum_{k=1}^{K} y_k/K$, and empirical covariance, $\hat{\Sigma}^2(\mathbf{y}) = \sum_{k=1}^{K}(y_k - \mu)(y_k - \mu)^T/K$. Taking conditional expectations of ω_{SL}, given θ, induces the Bayesian synthetic likelihood, $L_{\text{SL}}(\theta) = \mathbf{E}(\omega_{\text{SL}}(\theta, \mathbf{y}) \mid \theta)$, as an approximation of the likelihood, $\mathcal{L}(\theta)$.

A.3. Pseudo-marginal method

For the pseudo-marginal approach, introduced by Andrieu and Roberts [41], we suppose that there exists a simulation-based estimator, $\omega(\theta, \mathbf{y})$, such that the conditional expectation $L_\omega(\theta) = \mathbf{E}(\omega \mid \theta)$ is an *unbiased* estimate of $\mathcal{L}(\theta) = f(y_0 \mid \theta)$. For example, following Warne et al. [35], suppose that an intractable density $f(y \mid \theta)$ arising from a stochastic model can be decomposed into an underlying latent model, $x \sim g(\cdot \mid \theta)$, and an observation model, $y \sim h(\cdot \mid \theta, x)$, such that $f(y \mid \theta) = \int h(y \mid \theta, x) g(x \mid \theta) \mathrm{d}x$. Assume that the probability densities $h(y \mid \theta, x)$ of the observation model can be calculated. Then, for simulations $x_k \sim g(\cdot \mid \theta)$ of the latent model, where $k = 1, \ldots, K$, we can write

$$\omega(\theta, \mathbf{x}) = \frac{1}{K}\sum_{k} h(y_0 \mid \theta, x_k),$$

as a likelihood-free weighting. Taking expectations over $\mathbf{x} \sim g(\cdot \mid \theta)$, we have $L_\omega(\theta) = \mathbf{E}(\omega \mid \theta) = f(y_0 \mid \theta) = \mathcal{L}(\theta)$. Thus, $L_\omega(\theta)$ is an *exact approximation* [42].

Appendix B. Alternative conditional distributions for M

The proof above derives the performance measure $\mathcal{J}_{\text{mf}}$ given in Equation (9), under the assumption that the conditional distribution of M, given θ and $\mathbf{y}_{\text{lo}}$, is Poisson with mean $\mu(\theta, \mathbf{y}_{\text{lo}})$. The following corollaries adapt the expression for $\mathcal{J}_{\text{mf}}$ in the case of alternative conditional distributions for M. We first define the MSE,

$$E_{\text{mf}} = \iint \left[\lambda_{\text{hi}}(\theta, \mathbf{y}_{\text{lo}}) - \omega_{\text{lo}}(\theta, \mathbf{y}_{\text{lo}})\right]^2 \rho(\theta, \mathbf{y}_{\text{lo}}) \mathrm{d}\theta \mathrm{d}\mathbf{y}_{\text{lo}},$$

between $\omega_{\text{lo}}(\theta, \mathbf{y}_{\text{lo}})$ and $\lambda_{\text{hi}}(\theta, \mathbf{y}_{\text{lo}}) = \mathbf{E}(\omega_{\text{hi}} \mid \theta, \mathbf{y}_{\text{lo}})$.

Corollary 7. *If $M \sim \text{Bin}(M_{\max}, p(\theta, \mathbf{y}_{\text{lo}}))$ is binomially distributed with maximum value $M_{\max}$ and mean $\mu(\theta, \mathbf{y}_{\text{lo}})$, where $p = \mu/M_{\max}$, then*

$$\mathcal{J}_{\text{mf}}[\mu] = \left(c_{\text{lo}} + \iint \mu(\theta, \mathbf{y}_{\text{lo}}) c_{\text{hi}}(\theta, \mathbf{y}_{\text{lo}})\ \rho(\theta, \mathbf{y}_{\text{lo}}) \mathrm{d}\theta \mathrm{d}\mathbf{y}_{\text{lo}}\right) \times \left(V_{\text{mf}} - \frac{E_{\text{mf}}}{M_{\max}} + \iint \Delta_q(\theta)^2 \frac{\eta(\theta, \mathbf{y}_{\text{lo}})}{\mu(\theta, \mathbf{y}_{\text{lo}})}\ \rho(\theta, \mathbf{y}_{\text{lo}}) \mathrm{d}\theta \mathrm{d}\mathbf{y}_{\text{lo}}\right). \tag{B.1}$$

Proof. We substitute $\mathrm{Var}(M \mid \theta, \mathbf{y}_{\mathrm{lo}}) = \mu\left(1 - \mu/M_{\max}\right)$ into Equation (22), and the result follows. □

We note in the result above that for μ to be the conditional mean of $M \sim \mathrm{Bin}(M_{\max}, p(\theta, \mathbf{y}_{\mathrm{lo}}))$, we must constrain the values of μ such that $\mu(\theta, \mathbf{y}_{\mathrm{lo}}) \in (0, M_{\max}]$. This constraint alters the derivation of the optimal $\mu^\star$, in the case of a binomial conditional distribution with fixed $M_{\max}$.

Corollary 8. *If $M \sim \mathrm{Geo}(p(\theta, \mathbf{y}_{\mathrm{lo}}))$ is geometrically distributed on the non-negative integers, with mean $\mu(\theta, \mathbf{y}_{\mathrm{lo}})$, where $p = 1/(1+\mu)$, then*

$$\mathcal{J}_{\mathrm{mf}}[\mu] = \left(c_{\mathrm{lo}} + \iint \mu(\theta, \mathbf{y}_{\mathrm{lo}}) c_{\mathrm{hi}}(\theta, \mathbf{y}_{\mathrm{lo}})\ \rho(\theta, \mathbf{y}_{\mathrm{lo}}) \mathrm{d}\theta \mathrm{d}\mathbf{y}_{\mathrm{lo}}\right) \times \left(V_{\mathrm{mf}} + E_{\mathrm{mf}} + \iint \Delta_q(\theta)^2 \frac{\eta(\theta, \mathbf{y}_{\mathrm{lo}})}{\mu(\theta, \mathbf{y}_{\mathrm{lo}})}\ \rho(\theta, \mathbf{y}_{\mathrm{lo}}) \mathrm{d}\theta \mathrm{d}\mathbf{y}_{\mathrm{lo}}\right). \tag{B.2a}$$

Proof. We substitute $\mathrm{Var}(M \mid \theta, \mathbf{y}_{\mathrm{lo}}) = \mu(1+\mu)$ into Equation (22), and the result follows. □

Appendix C. Mean functions

Algorithm 4 Piecewise constant mean function $\mu_{\mathcal{D}}(\theta, y_{\mathrm{lo}}; \nu)$ used in MF-ABC Algorithm 3, depicted in Fig. 2c, at final iteration.

Require: $\theta = (k_1, k_2, k_3)$; $y_{\mathrm{lo}} = (y_1, y_2, \ldots, y_{10})$.
- **if** $y_7 \leq 13.867$ **then**
  - **return** $\nu_1 = 0.084$.
- **else**
  - **if** $k_3 \leq 1.14$ **then**
    - **if** $y_8 \leq 16.219$ **then**
      - **return** $\nu_2 = 0.616$.
    - **else**
      - **if** $k_3 \leq 0.88$ **then**
        - **return** $\nu_3 = 0.08$.
      - **else**
        - **if** $y_{10} \leq 26.208$ **then**
          - **if** $k_1 \leq 91.265$ **then**
            - **return** $\nu_4 = 0.313$.
          - **else**
            - **return** $\nu_5 = 0.76$.
          - **end if**
        - **else**
          - **if** $y_7 \leq 15.151$ **then**
            - **return** $\nu_6 = 0.761$.
          - **else**
            - **if** $y_1 \leq 3.371$ **then**
              - **if** $y_5 \leq 11.264$ **then**
                - **return** $\nu_7 = 0.688$.
              - **else**
                - **return** $\nu_8 = 0.467$.
              - **end if**
            - **else**
              - **return** $\nu_9 = 0.797$.
            - **end if**
          - **end if**
        - **end if**
      - **end if**
    - **end if**
  - **else**
    - **if** $y_{10} \leq 26.136$ **then**
      - **if** $y_7 \leq 14.17$ **then**
        - **return** $\nu_{10} = 0.929$.
      - **else**
        - **return** $\nu_{11} = 0.828$.
      - **end if**
    - **else**
      - **return** $\nu_{12} = 1.439$.
    - **end if**
  - **end if**
- **end if**

Algorithm 5 Piecewise constant mean function $\mu_{\mathcal{D}}(\theta, y_{\text{lo}}; \nu)$ used in MF-BSL Algorithm 3, depicted in Fig. 3c, at final iteration.

Require: $\theta = (k_1, k_2, k_3)$; $y_{\text{lo},i} = (y_{1,i}, y_{2,i}, \ldots, y_{10,i})$ for $i = 1, \ldots, 100$.
  if $y_{699} \leq 15.292$ **then**
    if $y_{779} \leq 16.237$ **then**
      if $y_{227} \leq 12.604$ **then**
        return $\nu_1 = 0.019$.
      else
        return $\nu_2 = 0.236$.
      end if
    else
      return $\nu_3 = 0.287$.
    end if
  else
    if $y_{115} \leq 9.86$ **then**
      if $y_{787} \leq 14.336$ **then**
        if $y_{446} \leq 9.953$ **then**
          return $\nu_4 = 0.222$.
        else
          if $y_{137} \leq 12.819$ **then**
            return $\nu_5 = 0.215$.
          else
            return $\nu_6 = 0.238$.
          end if
        end if
      else
        return $\nu_7 = 0.262$.
      end if
    else
      if $y_{889} \leq 22.689$ **then**
        if $y_{804} \leq 9.032$ **then**
          if $y_{981} \leq 1.974$ **then**
            return $\nu_8 = 0.229$.
          else
            return $\nu_9 = 0.175$.
          end if
        else
          if $y_{900} \leq 25.689$ **then**
            return $\nu_{10} = 0.235$.
          else
            return $\nu_{11} = 0.132$.
          end if
        end if
      else
        if $y_{209} \leq 24.088$ **then**
          return $\nu_{12} = 0.102$.
        else
          if $y_{390} \leq 30.822$ **then**
            return $\nu_{13} = 0.092$.
          else
            if $y_{180} \leq 35.199$ **then**
              return $\nu_{14} = 0.04$.
            else
              return $\nu_{15} = 0.003$.
            end if
          end if
        end if
      end if
    end if
  end if

References

[1] B. Peherstorfer, K. Willcox, M. Gunzburger, Survey of multifidelity methods in uncertainty propagation, inference, and optimization, SIAM Rev. 60 (2018) 550–591.
[2] B. Peherstorfer, K. Willcox, M. Gunzburger, Optimal model management for multifidelity Monte Carlo estimation, SIAM J. Sci. Comput. 38 (2016) A3163–A3194.
[3] L.W.T. Ng, K.E. Willcox, Multifidelity approaches for optimization under uncertainty, Int. J. Numer. Methods Eng. 100 (2014) 746–772.
[4] S.A. Sisson, Y. Fan, M. Beaumont (Eds.), Handbook of Approximate Bayesian Computation, CRC Press, 2020.
[5] M. Sunnåker, A.G. Busetto, E. Numminen, J. Corander, M. Foll, C. Dessimoz, Approximate Bayesian computation, PLoS Comput. Biol. 9 (2013) e1002803.
[6] L.F. Price, C.C. Drovandi, A. Lee, D.J. Nott, Bayesian synthetic likelihood, J. Comput. Graph. Stat. 27 (2018) 1–11.
[7] P. Marjoram, J. Molitor, V. Plagnol, S. Tavare, Markov chain Monte Carlo without likelihoods, Proc. Natl. Acad. Sci. 100 (2003) 15324–15328.
[8] S.A. Sisson, Y. Fan, M.M. Tanaka, Sequential Monte Carlo without likelihoods, Proc. Natl. Acad. Sci. 104 (2007) 1760–1765.
[9] T. Toni, D. Welch, N. Strelkowa, A. Ipsen, M.P. Stumpf, Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems, J. R. Soc. Interface 6 (2009) 187–202.

[10] P. Del Moral, A. Doucet, A. Jasra, An adaptive sequential Monte Carlo method for approximate Bayesian computation, Stat. Comput. 22 (2011) 1009–1020.
[11] T.P. Prescott, R.E. Baker, Multifidelity approximate Bayesian computation, SIAM/ASA J. Uncertain. Quantificat. 8 (2020) 114–138.
[12] T.P. Prescott, R.E. Baker, Multifidelity approximate Bayesian computation with sequential Monte Carlo parameter sampling, SIAM/ASA J. Uncertain. Quantificat. 9 (2021) 788–817.
[13] K. Cranmer, J. Brehmer, G. Louppe, The frontier of simulation-based inference, Proc. Natl. Acad. Sci. 117 (2020) 30055–30062.
[14] C.C. Drovandi, A.N. Pettitt, Estimation of parameters for macroparasite population evolution using approximate Bayesian computation, Biometrics 67 (2011) 225–233.
[15] D.J. Warne, T.P. Prescott, R.E. Baker, M.J. Simpson, Multifidelity multilevel Monte Carlo to accelerate approximate Bayesian parameter inference for partially observed stochastic processes, J. Comput. Phys. 469 (2022) 111543.
[16] N. Guha, X. Tan, Multilevel approximate Bayesian approaches for flows in highly heterogeneous porous media and their applications, J. Comput. Appl. Math. 317 (2017) 700–717.
[17] A. Jasra, S. Jo, D. Nott, C. Shoemaker, R. Tempone, Multilevel Monte Carlo in approximate Bayesian computation, Stoch. Anal. Appl. 37 (2019) 346–360.
[18] D.J. Warne, R.E. Baker, M.J. Simpson, Multilevel rejection sampling for approximate Bayesian computation, Comput. Stat. Data Anal. 124 (2018) 71–86.
[19] D.J. Warne, R.E. Baker, M.J. Simpson, Rapid Bayesian inference for expensive stochastic models, J. Comput. Graph. Stat. 31 (2021) 512–528.
[20] D. Prangle, A.B.C. Lazy, Stat. Comput. 26 (2016) 171–185.
[21] J.A. Christen, C. Fox, Markov chain Monte Carlo using an approximation, J. Comput. Graph. Stat. 14 (2005) 795–810.
[22] R.G. Everitt, P.A. Rowińska, Delayed acceptance ABC-SMC, J. Comput. Graph. Stat. 30 (2021) 55–66.
[23] L. Yan, T. Zhou, Adaptive multi-fidelity polynomial chaos approach to Bayesian inference in inverse problems, J. Comput. Phys. 381 (2019) 110–128.
[24] L. Yan, T. Zhou, An adaptive surrogate modeling based on deep neural networks for large-scale Bayesian inverse problems, Commun. Comput. Phys. 28 (2020) 2180–2205.
[25] J.J. Bon, D.J. Warne, D.J. Nott, C. Drovandi, Bayesian score calibration for approximate models, arXiv:2211.05357, 2022.
[26] D.T. Gillespie, Exact stochastic simulation of coupled chemical reactions, J. Phys. Chem. 81 (1977) 2340–2361.
[27] J. Bezanson, A. Edelman, S. Karpinski, V.B. Shah, Julia: a fresh approach to numerical computing, SIAM Rev. 59 (2017) 65–98.
[28] A.B. Owen, Monte Carlo theory, methods and examples, https://artowen.su.domains/mc/, 2013.
[29] M.B. Giles, Multilevel Monte Carlo methods, Acta Numer. 24 (2015) 259–328.
[30] C.-H. Rhee, P.W. Glynn, Unbiased estimation with square root convergence for SDE models, Oper. Res. 63 (2015) 1026–1043.
[31] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning, Springer, New York, 2009.
[32] D. Schnoerr, G. Sanguinetti, R. Grima, Approximation and inference methods for stochastic biochemical kinetics—a tutorial review, J. Phys. A, Math. Theor. 50 (2017) 093001.
[33] D.J. Warne, R.E. Baker, M.J. Simpson, Simulation and inference algorithms for stochastic biochemical reaction networks: from basic concepts to state-of-the-art, J. R. Soc. Interface 16 (2019) 20180943.
[34] R. Erban, S.J. Chapman, Stochastic Modelling of Reaction–Diffusion Processes, Cambridge University Press, 2019.
[35] D.J. Warne, R.E. Baker, M.J. Simpson, A practical guide to pseudo-marginal methods for computational inference in systems biology, J. Theor. Biol. 496 (2020) 110255.
[36] C. Lester, Multi-level approximate Bayesian computation, arXiv:1811.08866, 2019.
[37] M.E. Levine, A.M. Stuart, A framework for machine learning of model error in dynamical systems, arXiv:2107.06658, 2021.
[38] J.-M. Cornuet, J.-M. Marin, A. Mira, C.P. Robert, Adaptive multiple importance sampling, Scand. J. Stat. 39 (2012) 798–812.
[39] A.A. Gorodetsky, J.D. Jakeman, G. Geraci, MFNets: data efficient all-at-once learning of multifidelity surrogates as directed networks of information sources, Comput. Mech. 68 (2021) 741–758.
[40] D.T. Gillespie, Approximate accelerated stochastic simulation of chemically reacting systems, J. Chem. Phys. 115 (2001) 1716–1733.
[41] C. Andrieu, G.O. Roberts, The pseudo-marginal approach for efficient Monte Carlo computations, Ann. Stat. 37 (2009).
[42] C. Drovandi, R.G. Everitt, A. Golightly, D. Prangle, Ensemble MCMC: accelerating pseudo-marginal MCMC for state space models using the ensemble Kalman filter, Bayesian Anal. 17 (2022) 223–260.