Deep surrogate accelerated delayed-acceptance HMC for Bayesian inference of spatio-temporal heat fluxes in rotating disc systems

Teo Deveney*, Eike Mueller†, Tony Shardlow‡
Department of Mathematical Sciences, University of Bath, Bath, UK, BA2 7AY

Abstract

We study the Bayesian inverse problem of inferring the Biot number, a spatio-temporal heat-flux parameter in a PDE model. This is an ill-posed problem where standard optimisation yields unphysical inferences. We introduce a training scheme that uses temperature data to adaptively train a neural-network surrogate to simulate the parametric forward model. This approach approximates forward and inverse solution together, by simultaneously identifying an approximate posterior distribution over the Biot number, and weighting the forward training loss according to this approximation. Utilising random Chebyshev series, we outline how to approximate an arbitrary Gaussian process prior, and using the surrogate we apply Hamiltonian Monte Carlo (HMC) to efficiently sample from the corresponding posterior distribution. We derive convergence of the surrogate posterior to the true posterior distribution in the Hellinger metric as our adaptive loss function approaches zero. Furthermore, we describe how this surrogate-accelerated HMC approach can be combined with a traditional PDE solver in a delayed-acceptance scheme to a-priori control the posterior accuracy, thus overcoming a major limitation of deep learning-based surrogate approaches, which do not achieve guaranteed accuracy a-priori due to their non-convex training. Biot number calculations are involved turbomachinery design, which is safety critical and highly regulated, therefore it is important that our results have such mathematical guarantees. Our approach achieves fast mixing in high-dimensional parameter spaces, whilst retaining the convergence guarantees of a traditional PDE solver, and without the burden of evaluating this solver for proposals that are likely to be rejected. Numerical results compare the accuracy and efficiency of the adaptive and general training regimes, as well as various Markov chain Monte Carlo proposals strategies.

*email: T.J.Deveney@bath.ac.uk
†email: E.Mueller@bath.ac.uk
‡email: T.Shardlow@bath.ac.uk
1 Introduction

The disc temperature distribution in compressor cavities is a fundamental quantity of interest for aerospace engineers due to its effect on material expansion. In order to improve engine design, engineers are interested in simulating the temperature evolution over time, leading to an urgent requirement for accurate physical models of heat transfer inside the engine cavity. Appropriate parameterisations of these models in a transient setting are currently not well understood, and therefore experimental data can be extremely valuable in aiding our knowledge of the parameters. One very important parameter in such models is the Biot number, a function which can vary over space and time, that dictates the relative effects of convection and conduction on heat transfer.

Previous work has been carried out to infer the Biot number from temperature measurements in a stationary setting. In [2], polynomial curves were fit to data using least squares. This approach, while yielding a good fit, lacks appropriate regularisation which often leads to physically implausible inferences with large oscillations unless restricted to very low degree polynomials. This effect is highlighted in [17], where a Bayesian regularisation method is instead proposed based on maximum a posteriori (MAP) estimation over a spatial discretisation of the Biot number. This approach is shown to yield physical results and a local estimate of the uncertainty is achieved using a Laplace approximation based on the Hessian of the log-posterior. Using this Laplace approach, the full posterior distribution is not returned, meaning the uncertainty estimate may be unreliable. Furthermore, the increase in number of degrees of freedom of the Biot number coupled with the higher complexity of the PDE solve in the spatio-temporal setting pursued in this work, ensure that the extension of the approach of [17] to this case is computationally intractable. This is because it is reliant on numerically calculating the gradient of the posterior with respect to each degree of freedom of the discretised Biot number. More efficient approaches to achieve this are possible by using the adjoint PDE to solve the PDE-constrained optimisation problem, though a large number of numerical solves is still required in this case, and the local estimate of the associated uncertainty returned is insufficient as our numerical results reveal.

In this work, we overcome the limitations imposed by traditional numerical schemes in the spatio-temporal setting, by using deep learning to develop a Bayesian methodology capable of approximating the full posterior distribution of the Biot number. Our approach is to represent the parametric forward map by a neural network, thus greatly accelerating the simulation and differentiation of the PDE model. To attain this map efficiently, we design an adaptive training scheme, based on minimising the squared PDE-residual over a measure
that approximates the true posterior over the parameters. We demonstrate that this restriction to the posterior measure vastly improves the approximation speed and accuracy when compared to a general parametric approximation over a wider parameter space. Given our approximate forward map, we use the Hamiltonian Monte Carlo (HMC) sampling scheme [4] to generate proposal samples from the posterior distribution. Using this method, we are able to perform a full Bayesian analysis of the posterior distribution in minutes. Our results on simulated data show that a fully Bayesian approach is justified, as it provides a more accurate quantification of uncertainty than the Laplace approximation, which gives overly confident results for this problem.

PDE surrogates based on the approximation of parametric solutions by neural networks have been applied with impressive results previously [20, 21, 6]. However, due to the non-convex nature of the training procedure, these approaches suffer from an inability to guarantee the accuracy of the approximations that are attained. To overcome this, we additionally propose delayed-acceptance as part of our MCMC scheme [5]. In this delayed-acceptance HMC scheme, proposals which pass the initial surrogate-based Metropolis acceptance criteria are passed to a secondary acceptance criteria dependent on a Crank–Nicolson (CN) solver. The delayed-acceptance criteria is chosen such that the stationary distribution of the Markov chain satisfies detailed balance according to the likelihood induced by the CN solver. This approach, while slower than relying solely on the deep learning surrogate, ensures that the computation time dedicated to CN is being used optimally, since CN is only executed for proposals that have passed initial acceptance criteria and therefore have a high probability of acceptance, and successive proposals are decorrelated by the Hamiltonian proposal distribution. Furthermore, the CN solver is a well-studied space-time discretisation of the PDE solution with a rich convergence theory and quantifiable error [18]. As a result, we obtain a posterior sample that has the accuracy and convergence guarantees associated with a CN solver, but at a significantly lower computational cost than is possible by using CN in a typical Metropolis–Hastings sampler.

The remainder of this work proceeds as follows. In Section 2, we fully outline the Bayesian inverse problem for the Biot number that we consider throughout this work. Section 3 describes our methodology in various parts, beginning with how to specify a Gaussian process prior for the Biot number within a deep-learning surrogate, proceeded by a description of our adaptive training scheme, then the deep-surrogate-accelerated delayed-acceptance HMC sampling. In Section 4 we apply this methodology to simulated data. Our experiments compare the efficiency and accuracy of our adaptively trained surrogate to a surrogate trained over a more general set of parameters, and quantifies the statistical accuracy of different
sampling schemes in terms of the effective sample size (ESS) \cite{9} obtained. These experiments are carried out with and without the delayed-acceptance step, and the accuracy of the various sampling schemes and ‘surrogate only’ approaches (without delayed-acceptance) are visualised through posterior density plots.

2 Problem Specification

We consider heat transfer in rotating disc systems. To model the disc, we make an axisymmetric assumption in one spatial dimension representing radial location, and consider the evolution of the heat profile over time. After non-dimensionalisation, an appropriate PDE for the temperature of the disc is the transient fin equation

$$c_0 \frac{\partial u}{\partial t}(t, x) = c_1 \frac{\partial^2 u}{\partial x^2}(t, x) + c_2 \frac{\partial u}{\partial x}(t, x) - Bi(t, x)u(t, x), \quad t \in [0, T], \, x \in [a, b],$$

(2.1)

for \(0 < a < b\) and fixed parameters \(c_0, c_1, c_2 > 0\). To this equation, we prescribe Dirichlet boundary conditions

$$u(t, a) = u_a(t), \quad t \in [0, T],$$

$$u(t, b) = u_b(t), \quad t \in [0, T],$$

(2.2)

and an initial condition

$$u(0, x) = u_0(x), \quad x \in [a, b].$$

(2.3)

Given data \(\{(\hat{t}_n, \hat{x}_n, \hat{z}_n) : n = 1, 2, \ldots, N\}\), our goal is to perform Bayesian uncertainty quantification for the spatio-temporal parameter \(Bi(t, x)\) in (2.1) known as the Biot number. Here \((\hat{t}_n, \hat{x}_n)\) represent space-time coordinates and \((\hat{z}_n)\) represent temperature measurements corresponding to these coordinates.

We assume the data is related to the PDE through the statistical model

$$\hat{z}_n = u(\hat{t}_n, \hat{x}_n) + \epsilon_n, \quad n = 1, \ldots, N,$$

(2.4)

where \(u(\cdot, \cdot)\) is the solution to (2.1) and the noise terms \(\epsilon_n \sim N(0, \sigma^2_\epsilon)\) are i.i.d. Gaussian random variables with unknown standard deviation \(\sigma_\epsilon\). Under this model, the probability of observing the data for fixed parameters \(Bi(t, x), \sigma_\epsilon\) is given by the likelihood function

$$p(\hat{z} | \hat{t}, \hat{x}, Bi, \sigma_\epsilon) = \frac{1}{(2\pi\sigma^2_\epsilon)^{N/2}} \exp \left( -\frac{1}{2\sigma^2_\epsilon} \sum_{n=1}^{N} (\hat{z}_n - u(\hat{t}_n, \hat{x}_n))^2 \right).$$

(2.5)
To perform Bayesian inference, we must define a prior distribution \( p(B_i, \sigma) \) over the unknown parameters \( B_i(t, x) \) and \( \sigma \). The choice of prior distribution is important since it can be used as a regulariser to ensure that we recover physically meaningful results. Here we will assume that \( B_i(t, x) \) and \( \sigma \) are independent in the prior distribution, so that \( p(B_i, \sigma) = p_B(B_i)p_\sigma(\sigma) \) and we define \( p_\sigma(\sigma) \) to be a Gamma distribution. For \( p_B(B_i) \), we require the prior distribution to be defined over the space of 2-dimensional functions, and for this we choose the popular option of assigning a Gaussian process prior

\[
B_i \sim \mathcal{GP}(\mu(\cdot), K(\cdot, \cdot)).
\]

Here the mean function \( \mu(\cdot) \) and covariance kernel \( K(\cdot, \cdot) \) are manually specified such that they represent our prior beliefs about the behaviour of the function. In this work, we discretise the Gaussian process by a finite random-series expansion and the infinite-dimensional prior \( p_B(B_i) \) is represented explicitly by a density over finitely many discretisation parameters.

Having decided the prior distribution and likelihood [2.5], an application of Bayes rule gives us the posterior distribution

\[
p(B_i, \sigma | \hat{t}, \hat{x}, \hat{z}) = \frac{p(B_i, \sigma)p(\hat{z}|\hat{t}, \hat{x}, B_i, \sigma)}{p(\hat{z}|\hat{t}, \hat{x})} \propto p(B_i, \sigma)p(\hat{z}|\hat{t}, \hat{x}, B_i, \sigma).
\]

Approximating the posterior distribution is the aim of the Bayesian inverse problem. This measure represents the full conditional distribution of all unknown parameters given the observed data and prior information. Analysis of this distribution provides a robust and natural form of uncertainty quantification over the parameters, whilst also providing a complete picture of any correlations, skews, or heavy tails present in the distribution that might be important when assessing the results in practice.

## 3 Methodology

Our methodology broadly consists of 3 stages:

1. Represent \( B_i(t, x) \) by a series expansion and compute the distribution over the coefficients such that the series approximates a Gaussian process prior.

2. Train a deep learning surrogate model to approximate the parametric forward problem on an appropriate measure over the series coefficients.

3. Apply HMC within a Crank–Nicolson delayed-acceptance scheme to accurately sample from the posterior distribution using the computed prior distribution and deep learning surrogate.
Here we describe each of these steps in detail, outlining various options and their relative benefits in practice.

3.1 Prior approximation

We represent the functional parameter $B_i(t, x)$ through the Chebyshev expansion

$$
\hat{B}_i(t, x) = \sum_{i=1}^{M} \alpha_i T_i(t, x).
$$

(3.1)

In this expression $T_i(t, x)$ is a two dimensional Chebyshev polynomial basis function of degree less than or equal to $D$. $M = \frac{1}{2}(D + 1)(D + 2)$ is the number of basis terms, and for each $k, l \in \{0, 1, \ldots, D\}$ such that $k + l \leq D$ there is a unique $i \in \{1, \ldots, M\}$ such that $T_i(t, x) = T_k(t)T_l(x)$, where $T_k(t), T_l(x)$ are the shifted Chebyshev polynomials of the first kind onto the domains $t \in [0, T]$ and $x \in [a, b]$ respectively. The coefficients $\alpha \in \mathbb{R}^M$ are parameters that we wish to infer using data, and will be given as inputs to the deep learning surrogate model. Given this representation of $B_i(t, x)$ as a linear basis expansion, we seek to define a meaningful prior distribution over the coefficients of this expansion. The Gaussian process class of distributions is a natural target, as it allows us to define prior distributions over functions in a way that is easily interpretable through the specification of the mean function and covariance kernel. We ensure that $\hat{B}_i(t, x)$ is itself a Gaussian process by assigning a multivariate normal prior over the coefficients $\alpha \sim \text{MVN}(m, \Sigma)$,

$$
\alpha \sim \text{MVN}(m, \Sigma),
$$

(3.2)

where $m \in \mathbb{R}^M$ and $\Sigma \in \mathbb{R}^{M \times M}$ are the mean vector and covariance matrix.

The mean function of $\hat{B}_i(t_j, x_j)$ is straightforward to calculate as

$$
\hat{\mu}(t, x) = \sum_{i=1}^{M} m_i T_i(t, x),
$$

(3.3)

and the covariance function is

$$
\hat{K}([t, x], [t', x']) = \sum_{i,j=1}^{M} T_i(t, x)\Sigma_{i,j}T_j(t', x').
$$

(3.4)

We can then approximate a Gaussian process using (3.1) by approximating the desired deterministic mean and covariance using (3.3, 3.4). More explicitly, suppose that we wish to approximate the Gaussian process $B_i(t, x) \sim \mathcal{GP}(\mu(t, x), K([t, x], [t', x']))$; we can compute the mean vector $(m_1, \ldots, m_M)$ such that

$$
\sum_{i=1}^{M} m_i T_i(t, x) \approx \mu(t, x),
$$

(3.5)
and the covariance matrix entries \((\Sigma_{1,1}, \Sigma_{1,2}, \ldots, \Sigma_{M,M})\) such that
\[
\sum_{i,j=1}^{M} T_i(t,x)\Sigma_{i,j}T_j(t',x') \approx K([t,x],[t',x']).
\] (3.6)

The computation of \(\mathbf{m}\) and \(\Sigma\) are function approximation problems that are achieved by interpolation on Chebyshev nodes. This is guaranteed to be close to the optimal polynomial in the maximum norm for any continuous kernel function, and is computationally efficient due to its reliance only on interpolation [19]. Another approach that we could apply is the Karhunen–Loève (KL) expansion [12]. In the KL expansion the basis functions and the distribution of the coefficients are solutions to an eigenvalue problem, and the corresponding approximation is the best linear approximation to the Gaussian field in terms of the mean-squared error. We note that while the KL expansion defines a more accurate approximation to the GP, it is inconvenient in practice as it requires us to train a new surrogate with different basis functions each time we change the prior distribution. By instead taking the approach described above, we require only one surrogate based on Chebyshev polynomials, and if we wish to change the prior we need only recompute mean vector and covariance matrix of the random vector \(\alpha\).

The resultant distribution of \(\hat{B}_i(t,x)\) is a Gaussian process with mean function and covariance kernel approximately equal to those of the original Gaussian process \(B_i(t,x)\), however it is expressed entirely by a multivariate normal distribution over the coefficients \(p_\alpha(\alpha)\), with mean and covariance according to (3.2). We can subsequently use this as the prior distribution for our method. The corresponding posterior distribution is therefore
\[
p(\alpha, \sigma|\hat{t}, \hat{x}, \hat{z}) = \frac{p(\alpha, \sigma)p(\hat{z}|\hat{t}, \hat{x}, \alpha, \sigma)}{p(\hat{z}|\hat{t}, \hat{x})} \propto p(\alpha, \sigma)p(\hat{z}|\hat{t}, \hat{x}, \alpha, \sigma),
\] (3.7)
where \(p(\alpha, \sigma) = p_\alpha(\alpha)p_\sigma(\sigma)\), and \(p(\hat{z}|\hat{t}, \hat{x}, \alpha, \sigma)\) is the likelihood function
\[
p(\hat{z}|\hat{t}, \hat{x}, \alpha, \sigma) = \frac{1}{(2\pi\sigma^2)^{N/2}} \exp\left(-\frac{1}{2\sigma^2} \sum_{n=1}^{N} (\hat{z}_n - u(\hat{t}_n, \hat{x}_n, \alpha))^2\right).
\] (3.8)

Here \(u(\hat{t}, \hat{x}, \alpha)\) is the parametric solution to the PDE over the coefficients \(\alpha\), which will be approximated by a deep surrogate model in our method.

### 3.2 Deep surrogate approximation

Now we turn to approximating the parametric solution to the fin equation (2.1) over the parameters \(\alpha\). Let us define the interior PDE domain \(\Omega = [0, T] \times [a, b]\), and associate to it a positive measure \(\pi^\Omega\). Similarly define the boundary domain \(\partial\Omega = [0, T] \times \{a, b\} \cup \{0\} \times [a, b]\)
with positive measure $\pi^b$, and note that the boundary and initial conditions (2.2, 2.3) can be combined into a single Dirichlet condition

$$u(t, x) = u_{BC}(t, x), \quad (t, x) \in \partial \Omega.$$  \hfill (3.9)

To construct the deep surrogate model, we approximate the parametric solution to (2.1) using a neural network $\hat{u}(t, x, \alpha) : \Omega \times \mathbb{R}^M \rightarrow \mathbb{R}$. We associate a positive measure $\pi^\alpha$ on $\mathbb{R}^M$ to these coefficients, and train the network using stochastic gradient descent to minimise the loss function

$$\text{Loss} = \int_{\mathbb{R}^M} F(\alpha) \, d\pi^\alpha(\alpha),$$

$$F(\alpha) = \nu_1 \|L \hat{u}(\cdot, \cdot, \alpha) - b(\cdot, \cdot)\|^2_{L_2(\Omega, \pi)} + \nu_2 \| \hat{u}(\cdot, \cdot, \alpha) - u_{BC}(\cdot, \cdot, \alpha)\|^2_{L_2(\partial \Omega, \pi^b)},$$  \hfill (3.10)

for some parameters $\nu_1, \nu_2 > 0$. Here $b = 0$, and $L \hat{u}$ is the differential operator of the fin equation applied to the surrogate

$$L \hat{u}(t, x, \alpha) = \frac{\partial^2 \hat{u}}{\partial x^2}(t, x, \alpha) + \frac{1}{x} \frac{\partial \hat{u}}{\partial x}(t, x, \alpha) - B_i(t, x) \hat{u}(t, x, \alpha) - \frac{\partial \hat{u}}{\partial t}(t, x, \alpha).$$  \hfill (3.11)

We use the subscript $L_2(C, \mu)$ notation to denote the $L_2$-norm over domain $C$ with respect to the measure $\mu$

$$\|g\|^2_{L_2(C, \mu)} = \int_C |g(x)|^2 \, d\mu(x).$$  \hfill (3.12)

The integrals in (3.10) are intractable, so in practice this minimisation is implemented by drawing randomised collocation points $(t, x, \alpha)$ from the interior measure $\pi \otimes \pi^\alpha$ and boundary points $(t_b, x_b, \alpha_b)$ from $\pi^b \otimes \pi^\alpha$, then minimising the Monte Carlo approximation of the integral in (3.12) induced by these points. This reduction is achieved by taking a gradient descent step, and after each step a new random sample of collocation points is drawn and the process repeated. This approach is an extension of the Deep Galerkin Method first introduced for static parameter values in [15].

We will take $\pi$ and $\pi^b$ to be uniform on $\Omega$ and $\partial \Omega$ respectively throughout this work. The parameter measure $\pi^\alpha$ however is a key ingredient whose specification can have a significant impact on both the accuracy the surrogate and the efficiency of its training. It has been shown that the number of neural network parameters required to accurately approximate the parametric solution to several classes of PDE depends only on the intrinsic dimension of the solution manifold [10]. This is in contrast to a dependency on the dimensionality of PDE parameters themselves, meaning solutions approximated using this method are capable of overcoming the curse of dimensionality associated with large numbers of parameters if the
solution over the training domain has a low intrinsic dimension. The measure $\pi^\alpha$ influences the intrinsic dimensionality of the solution manifold and so its specification is important.

Previous work has set $\pi^\alpha$ to be a uniform distribution on a compact subset of $\mathbb{R}^M$ [6], producing a general surrogate spanning a relatively large parameter space. This has the advantage that the surrogate need only be trained once, resulting in an analytic function that can be stored and applied to various datasets, however this surrogate will typically be expensive to train and less accurate than a more focused measure. In Section 4.2, we demonstrate that this is the case in our setting, and that using a general surrogate leads to inaccurate inferences for this problem. An alternative approach is to set $\pi^\alpha$ to be the posterior distribution. By definition, this measure weights the training across parameter space exactly as we require, and since this is concentrated around parameters that achieve a good fit to the data the corresponding solution manifold will have a lower intrinsic dimension. Of course this approach assumes that we know the posterior distribution a-priori, which is not the case in practice, and thus requires an adaptive training regime that trains the surrogate and approximates the posterior simultaneously.

**Data adaptive training**

Our approach to training the surrogate seeks to set the parameter measure $\pi^\alpha$ in the loss function (3.10) to be the posterior distribution (3.7). This is ultimately achieved by using MCMC samples from the posterior as training points for the surrogate. However, this approach raises the issue of how to initially generate these samples given that we require a trained surrogate to begin MCMC. Our solution is to first train the surrogate over the Laplace approximation to the posterior distribution

$$\pi^\alpha_{\text{Laplace}} = MVN(\alpha^*, \text{Hess}_\alpha^{-1} \{-\log p(\alpha^*, \sigma^* | \hat{t}, \hat{x}, \hat{z})\}),$$

(3.13)

where $\alpha^*, \sigma^*$ are the MAP estimates

$$(\alpha^*, \sigma^*) = \text{argmax} \{ \log p(\alpha, \sigma | \hat{t}, \hat{x}, \hat{z}) \}.$$  

(3.14)

The MAP optimisation is carried out by beginning with some initial estimate $(\alpha_0, \sigma_{0\ell})$ and iteratively updating this estimate using gradient ascent. To approximate the gradients to the objective function (3.14) at iteration $n$, the deep surrogate approximation is updated to approximate the solution over a local radius surrounding $\alpha_n$ and automatically differentiated. These gradients are applied to update to $\alpha_{n+1}$, and the process is repeated. The size of the local radius and the step size is reduced as we approach the MAP estimate in a manner similar to trust region optimisation (see the appendix for more detail). Given
the MAP estimate \((\alpha^*, \sigma^*_\epsilon)\) and the corresponding locally trained surrogate \(\hat{u}^*(t, x, \alpha)\), the
covariance of the Laplace approximation to the posterior distribution is readily available by
automatic differentiation, thus we next train a surrogate over this approximation by using
samples from the Laplace approximation as training points. This initial surrogate is then
utilised within an MCMC scheme to begin sampling from the posterior distribution.

Although the Laplace approximation provides a reasonable estimate of the posterior dis-
tribution for many applications, it may be inaccurate if the true posterior deviates too far
from a Gaussian. To account for this and ensure that the training measure \(\pi^\alpha\) is as close
as possible to the posterior distribution, we continue training the surrogate online using
posterior samples from the MCMC scheme as training points for the loss function (3.10).
This additional refinement of the approximation is interleaved with the MCMC sampler
during a warm-up period, after which the surrogate is fixed and used to sample from the
posterior distribution.

We can understand the relation between the training loss function and the accuracy of the
posterior distribution induced by the adaptively trained surrogate in the Hellinger metric
as follows. Let us denote the true posterior by \(\psi(\theta)\) and let \(\psi_0(\theta)\) be the prior distribution
so that

\[
d\psi = \frac{1}{Z} \exp(-\Phi(\theta)) d\psi_0(\theta).
\] (3.15)

Here \(\theta = (\alpha, \sigma_\epsilon)\), \(\Phi(\theta) = \frac{1}{2\sigma_\epsilon^2} \sum_{n=1}^{N}(\hat{z}_n - u(\hat{t}_n, \hat{x}_n, \alpha))^2\) and \(Z\) is a normalisation constant.
Denote the approximated posterior induced by substituting the surrogate into the likelihood
by \(\hat{\psi}(\theta)\), and its associated potential by \(\hat{\Phi}(\theta)\).

Let \(\frac{d\psi}{d\psi_0} = f\) and \(\frac{d\hat{\psi}}{d\psi_0} = g\) be Radon–Nikodym derivatives with respect to \(\psi_0\). Then the
squared Hellinger distance between \(\psi\) and \(\hat{\psi}\) is

\[
H^2(\psi, \hat{\psi}) = \frac{1}{2} \int (\sqrt{f(\theta)} - \sqrt{g(\theta)})^2 d\psi_0(\theta)
= \frac{1}{2} \int (\sqrt{f(\theta)/g(\theta)} - 1)^2 d\hat{\psi}(\theta).
\] (3.16)

Assuming \(\Phi(\theta) \approx \hat{\Phi}(\theta)\) and \(Z \approx \hat{Z}\), we can approximate

\[
H^2(\psi, \hat{\psi}) \approx \frac{1}{8} \int \left( \frac{\sqrt{\hat{Z}}}{\sqrt{Z}} \exp \left( \frac{1}{2}(\hat{\Phi}(\theta) - \Phi(\theta)) \right) - 1 \right)^2 d\hat{\psi}(\theta)
\approx \frac{1}{8} \int (\hat{\Phi}(\theta) - \Phi(\theta))^2 d\hat{\psi}(\theta).
\] (3.17)
By assuming that $u$ and $\hat{u}$ are bounded uniformly, we have for some $M > 0$ that

$$|\Phi(\theta) - \hat{\Phi}(\theta)| = \left| \frac{1}{2\sigma^2} \sum_{n=1}^{N} (\hat{u}(\hat{t}_n, \hat{x}_n, \alpha) - u(\hat{t}_n, \hat{x}_n, \alpha)) (\hat{u}(\hat{t}_n, \hat{x}_n, \alpha) + u(\hat{t}_n, \hat{x}_n, \alpha) - 2\hat{z}_n) \right| \leq \frac{M}{2\sigma^2} \sum_{n=1}^{N} |(\hat{u}(\hat{t}_n, \hat{x}_n, \alpha) - u(\hat{t}_n, \hat{x}_n, \alpha))|,$$

using $(a - c)^2 - (b - c)^2 = (a - b)(a + b - 2c)$. So applying Jensen’s inequality we have that

the Hellinger distance approximately satisfies

$$H^2(\psi, \hat{\psi}) \leq \frac{NM^2}{32} \int \frac{1}{\sigma^4} \sum_{n=1}^{N} (\hat{u}(\hat{t}_n, \hat{x}_n, \alpha) - u(\hat{t}_n, \hat{x}_n, \alpha))^2 d\hat{\psi}(\theta).$$

(3.19)

Now, $\hat{u}$ satisfies a second-order PDE similar to (2.1) with perturbed initial and boundary data and an extra forcing term (by substituting $\hat{u}$ into (2.1)). We may apply $L^2$-stability theory for second-order parabolic equations (e.g., [14]), to determine that,

$$\frac{1}{N} \sum_{n=1}^{N} (\hat{u}(\hat{t}_n, \hat{x}_n, \alpha) - u(\hat{t}_n, \hat{x}_n, \alpha))^2 \approx \frac{1}{A} \|\hat{u}(\cdot, \cdot, \alpha) - u(\cdot, \cdot, \alpha)\|^2_{L^2(\Omega)} \leq F(\alpha),$$

(3.20)

where $F$ is the size of the perturbed boundary data and extra forcing term as defined in (3.10) with appropriate weighting $\nu_1, \nu_2$. Here we have approximated $\frac{1}{N} \sum_{n=1}^{N} (\hat{u}(\hat{t}_n, \hat{x}_n, \alpha) - u(\hat{t}_n, \hat{x}_n, \alpha))^2$ by $\frac{1}{A} \|\hat{u} - u\|^2_{L^2(\Omega)}$, where $A$ is the area of $\Omega$, which is reasonable in our settings where data is collected at regular space-time intervals. Then, for some $C > 0$,

$$H^2(\psi, \hat{\psi}) \lesssim C \int \frac{1}{\sigma^4} F(\alpha) d\hat{\psi}(\alpha, \sigma).$$

(3.21)

The right-hand side corresponds to the training loss function (3.10) when the prior on the hyper-parameter $\sigma$ is trivial ($\sigma$ is treated as known) and describes the appropriate generalisation of (3.10) to non-trivial priors on $\sigma$. When evaluated with MCMC samples from the surrogate posterior, therefore, it provides an approximate bound on the posterior error in the Hellinger metric. A rigorous treatment that quantifies the effect of the $\Phi(\theta) \approx \hat{\Phi}(\theta)$ and $Z \approx \hat{Z}$ assumptions can also be achieved following the methodology in Chapter 4 of [16].

3.3 Approximating the posterior using delayed-acceptance HMC

The main aim in a Bayesian inverse problem is to infer the posterior distribution. Most commonly for PDE-based problems these approaches will be based upon optimisation of parameterised approximations to the target distribution. Such methods — which include
variational techniques and Laplace approximations — have the advantage of being numerically tractable using traditional techniques for PDEs. However this approximation to the target distribution is based upon assumptions made about its shape a-priori. Alternatively, Markov chain Monte Carlo methods empirically approximate the target by sampling directly from it. The resulting approximations can be made arbitrarily accurate by the law of large numbers, but require a large number of decorrelated samples. This requirement has restricted the application of MCMC to PDE-based inverse problems as each MCMC iteration requires the PDE to be solved numerically, which is an expensive undertaking.

The major advantages of deep neural network approximations to the parametric solutions of PDEs is that they can efficiently be evaluated and differentiated with respect to the PDE parameters using automatic differentiation within dedicated software such as TensorFlow[1]. This ease of manipulation of the deep surrogate model affords us great flexibility in how we approximate the posterior distribution. For example, one can quickly establish a Laplace approximation as described in Section 3.2 or use their differentiability to efficiently implement gradient-based MCMC schemes such as Hamiltonian Monte Carlo (HMC). Conversely, a current disadvantage of deep learning-based PDE solvers is a lack of convergence guarantees. Despite the remarkable approximation capabilities of neural networks for many classes of PDEs, our ability to realise these approximations typically relies on non-convex optimisation using stochastic gradient descent, and as a result we are currently unable to guarantee the accuracy of deep learning surrogates.

In the following we introduce the sampling scheme that we will apply to infer $\hat{Bi}(t,x)$. Our scheme efficiently combines the gradient-based proposal distributions defined by the Hamiltonian Monte Carlo method with the guaranteed accuracy of a Crank–Nicolson solver using a delayed-acceptance verification step. We begin by describing Metropolis–Hastings MCMC, and then highlight how a Hamiltonian delayed-acceptance proposal distribution can be incorporated to improve sampling efficiency and accuracy.

**Metropolis–Hastings Markov chain Monte Carlo**

A Markov chain is a discrete-time stochastic process $X_0, X_1, X_2, \ldots$ such that $X_{n+1}$ given $X_n$ is independent of all other previous states. In our application we wish to sample from the posterior distribution over the parameters $\theta = (\alpha, \sigma)$, accordingly let us consider a stochastic process $\theta_0, \theta_1, \theta_2, \ldots$ taking values in the parameter space $\Theta \subset \mathbb{R}^M \times \mathbb{R}^+$. If the process satisfies the Markov property

$$g(\theta_{n+1}|\theta_n, \ldots, \theta_0) = g(\theta_{n+1}|\theta_n)$$  \hspace{1cm} (3.22)
then it is a Markov chain, and \( g(\theta_{n+1} | \theta_n) \) is the transition kernel of the process (the probability density of transitioning from \( \theta_n \) to \( \theta_{n+1} \)). The stationary distribution of a Markov chain is a probability measure \( \psi \) such that \( \psi(\theta) \geq 0 \ \forall \theta \in \Theta \), \( \int_{\Theta} d\psi(\theta) = 1 \) and \( \int_B d\psi(\theta) = \int_{\Theta} g(B|\theta) d\psi(\theta) \) for any \( B \subset \Theta \). Here the left hand side is the probability of \( B \), whereas the right side is the probability that transitions into \( B \) from \( \psi \) in one time step. Their equality implies that \( \psi \) remains constant after transitioning.

Markov chain Monte Carlo methods are a group of algorithms for sampling from a distribution known up to an arbitrary constant of proportionality. This is achieved by simulating ergodic Markov chains for which the target distribution is their stationary distribution. The states visited by the process then form a sample from this distribution and the empirical density of these states converges in distribution to the target. A sufficient condition for constructing an appropriate Markov chain is detailed balance, which states that \( g(\theta'|\theta)\psi(\theta) = g(\theta|\theta')\psi(\theta') \) holds for any \( \theta, \theta' \in \Theta \). It can be shown that if \((g, \psi)\) satisfies detailed balance, then \( \psi \) is the stationary distribution of the process. In MCMC methods we construct transition kernels that achieve detailed balance for a given distribution. In particular if we choose the target distribution as \( \psi(\theta) = p(\theta | \hat{\ell}, \hat{x}, \hat{z}) \) defined in (3.7) then this allows us to sample from the posterior. One of the simplest approaches to the construction of appropriate transition kernels is the Metropolis–Hastings (MH) algorithm.

In the Metropolis–Hastings algorithm we choose a proposal density \( q(\theta_{\text{prop}} | \theta) \) describing the probability of proposing a transition from \( \theta \) to \( \theta_{\text{prop}} \). This proposal is then accepted with probability

\[
A_M(\theta_{\text{prop}}, \theta) = \min \left\{ 1, \frac{q(\theta | \theta_{\text{prop}}) \psi(\theta_{\text{prop}})}{q(\theta_{\text{prop}} | \theta) \psi(\theta)} \right\}.
\]

(3.23)

It is readily verifiable that detailed balance is preserved with respect to \( \psi \) if the transitions are carried out this way. The full algorithm can be written as:
Algorithm 1: Metropolis-Hastings

Choose initial $\theta_0 \in \Theta$;

for $i = 0, 1, \ldots, N_{\text{samples}}$ do

- Propose $\theta_{\text{prop}} \sim q(\theta_{\text{prop}}|\theta_i)$;
- Sample $u \sim U(0, 1)$;
- if $u \leq A_M(\theta_{\text{prop}}, \theta_i)$; then
  - set $\theta_{i+1} := \theta_{\text{prop}}$ (accept);
- else
  - set $\theta_{i+1} := \theta_i$ (reject);

end

return $\theta = (\theta_0, \theta_1, \ldots, \theta_{N_{\text{samples}}})$

The choice of proposal distribution $q(\theta_{\text{prop}}|\theta)$ is key to the success of MH. An efficient proposal distribution will yield samples that are not heavily correlated, resulting in a sample that estimates the posterior with a lower statistical error. For a sample generated using some proposal distribution, the effective sample size (ESS) quantifies the efficiency of the proposal by returning the number of i.i.d samples required to reach the same level of precision [9]. In practice simple symmetric proposal distributions such as the multivariate Gaussian distribution centred at $\theta$ are commonly used. This random walk Metropolis–Hastings (RWMH) scheme has the advantage of simplicity, as it requires just one PDE solve per iteration and results in an acceptance probability that is independent of the proposal distribution. Unfortunately exploration strategies based on symmetric random walks such as this typically result in Markov chains that have a low acceptance rate and highly correlated samples in higher dimensions, leading to a low ESS and slow convergence to the stationary distribution.

Alternative approaches — such as Hamiltonian Monte Carlo (HMC) and Metropolis Adjusted Langevin (MALA) — use gradients of the target distribution to improve the efficiency of MCMC. For brevity we reserve a detailed treatment of HMC to the appendix. Put briefly, in HMC we define a Hamiltonian function using the target distribution by

$$H(\theta, p) = -\log \psi(\theta) - \log \psi(p|\theta)$$

$$= U(\theta) + K(p|\theta), \quad (3.24)$$

Here $U(\theta) = -\log \psi(\theta)$ represents the potential energy and $K(p|\theta) = -\log \psi(p|\theta)$ represents the kinetic energy of the system, whereas $\psi(p|\theta)$ is a user specified distribution over some auxiliary momentum variables. Proposals are then generated by using symplectic integrators to simulate random trajectories from the corresponding Hamiltonian system. This has the effect of significantly reducing the autocorrelation of the chain. It is impracti-
cal to perform HMC using traditional PDE solvers since $\nabla_\theta \psi(\theta)$ is required at each step of the symplectic integrator, leading to tens or hundreds of PDE solves being performed for each MCMC sample. MALA, by comparison, simulates the Langevin dynamics of a process that has stationary distribution $\psi(\theta)$, and can be shown to be equivalent to performing HMC with trajectories simulated over just one time step. MALA therefore represents a more tractable intermediary between random walk and Hamiltonian proposals, requiring just two gradients per iteration which can be calculated using adjoint PDE methods. This yields some improvement over symmetric random walk proposals, however we demonstrate that it is significantly outperformed by HMC, thus justifying the need for efficient HMC methods for PDEs.

**Delayed-acceptance HMC**

With deep surrogate models the burden of repeated differentiation of the PDE solution is alleviated by automatic differentiation, therefore HMC can be performed efficiently. Unfortunately the accuracy of deep surrogate solutions cannot be mathematically guaranteed as is the case with traditional numerical solvers such as Crank–Nicolson. This inaccuracy has the potential to bias the posterior sample, leading to unreliable estimates that cannot safely be used in turbine design. The delayed-acceptance HMC sampler that we describe here overcomes this by performing Metropolis–Hastings using a Crank–Nicolson solver with a proposal distribution constructed such that:

1. HMC proposals based on the deep learning surrogate are utilised for efficient exploration of the posterior distribution.

2. The Crank–Nicolson solver is only executed if the proposal is accepted according to the regular HMC acceptance criteria based on the surrogate.

The delayed-acceptance method was first described in [5]. To describe our adaptation of this method, let us first denote the posterior distribution induced by substituting the deep surrogate model $\hat{u}(\hat{t}_n, \hat{x}_n, \alpha)$ into the likelihood (3.8) by $\hat{\psi}(\theta)$, and similarly denote by $\tilde{\psi}(\theta)$ the posterior induced by the Crank–Nicolson solution $\tilde{u}(\hat{t}_n, \hat{x}_n, \alpha)$.

In the delayed-acceptance HMC scheme we sample from $\tilde{\psi}(\theta)$ using Metropolis–Hastings with proposal distribution given by

$$q(\theta_{\text{prop}}|\theta) = A_H(\theta_{\text{prop}}, \theta)q_H(\theta_{\text{prop}}|\theta)$$

$$+ \left(1 - \int (A_H(\theta_{\text{prop}}, \theta)q_H(\theta_{\text{prop}}|\theta)) \, d\theta_{\text{prop}}\right) \delta(\theta - \theta_{\text{prop}})$$

(3.25)
Here $q_H(\theta_{prop}, \theta)$ and $A_H(\theta_{prop}, \theta)$ are the HMC proposal distribution and acceptance probability applied using $\hat{\psi}(\theta)$. The first term in (3.25) is therefore the density of HMC proposals that are accepted, while the second term assigns the remaining probability to the chain remaining in place at $\theta$.

It is natural to execute this proposal in two stages. During the first, a regular HMC proposal is made with standard acceptance criteria (5.6), this acts as preliminary screening by the surrogate. If rejected by this criteria, the Markov chain remains in the same state and the Crank–Nicolson solver is not executed. If accepted, the proposal is passed to a secondary acceptance criteria with acceptance probability

$$A(\theta_{prop}, \theta) = \min \left\{ 1, \frac{\tilde{\psi}(\theta_{prop}) \hat{\psi}(\theta)}{\tilde{\psi}(\theta) \hat{\psi}(\theta_{prop})} \right\}. \quad (3.26)$$

This additional acceptance criteria ensures detailed balance is achieved with respect to $\tilde{\psi}$, thus the posterior is accurate to the level of the Crank–Nicolson solver. One can verify this by substituting (3.25) into the standard Metropolis acceptance criteria (3.23) and simplifying (see [8] for the full calculation).

The sampling efficiency and preliminary screening offered in the delayed-acceptance HMC algorithm ensures we are able to maximise the utility of the Crank–Nicolson solves. This allows us to tractably approximate the posterior by MCMC with the guaranteed accuracy that is critical in practice. There is of course some additional computational effort in comparison to relying purely on the surrogate, which we quantify in our simulation study below by comparing the performance of various sampling schemes based on surrogates with and without delayed acceptance steps.

4 Numerical results

The remainder of this work is dedicated to demonstrating deep surrogate accelerated delayed-acceptance HMC on simulated data. We first outline the specifics of our implementation, detailing the choice of prior, network architecture, training strategy and HMC calibration. Using simulated data we then demonstrate the speed and accuracy of our approach in comparison to alternative methods. All of the results and timings presented were produced in TensorFlow using a machine with a mobile RTX 2080 and a 6 core 3.9Ghz processor.
4.1 Implementation details

Prior distribution

We assign a Gaussian process prior distribution to $Bi(t, x)$ with mean equal to zero $\mu(t, x) = 0$, and a separable covariance kernel in space and time

$$K([t, x], [t', x']) = \sigma^2 K_x(x, x')K_t(t, t'). \quad (4.1)$$

For the spatial covariance $K_x(x, x')$ we adopt the twice differentiable Matérn kernel

$$K_x(x, x') = \left(1 + \frac{|x|}{\rho_x} + \frac{|x - x'|^2}{3\rho_x^2}\right) \exp\left(-\frac{|x - x'|}{\rho_x}\right). \quad (4.2)$$

Here we choose length scale of $\rho_x = b - a = 0.7$ as this is consistent with the spatial lengthscales applied in the stationary case [17]. For the temporal covariance we use the squared exponential kernel

$$K_t(t, t') = \exp\left(-\frac{|t - t'|^2}{2\rho_t^2}\right). \quad (4.3)$$

Here we choose $\rho_t = 900$, representing a temporal correlation length-scale of 15 minutes.

The marginal standard deviation of the prior distribution is $\sigma = 100$. Using (3.5, 3.6) the corresponding prior distribution over the coefficients is $\alpha \sim MVN(0, \Sigma)$ where $\Sigma$ is chosen such that

$$\sum_{i,j=1}^{M} T_i(t, x)\Sigma_{i,j} T_j(t', x') \approx K([t, x], [t', x']). \quad (4.4)$$

We resolve this expression using interpolation on the grid of Chebyshev nodes. The accuracy of this approximation is visualised for each of the spatial and temporal kernel factors in Figure [1].
Figure 1: Left: Chebyshev approximation to $K_x(x, x')$ and $K_t(t, t')$. Right: Error between the true and approximated kernel.

**Network architecture**

We use a fully-connected feed-forward neural network consisting of 4 hidden layers of 256 neurons with tanh activation functions to approximate the parametric solution. This deep learning surrogate $\hat{u}: \mathbb{R}^{68} \to \mathbb{R}$ takes as inputs the coordinates $(t, x) \in \mathbb{R}^2$, and the Chebyshev polynomial coefficients $\alpha \in \mathbb{R}^{66}$, and returns an approximation to the corresponding PDE solution at those coordinates.

**Training**

To train the deep learning surrogate we minimise the loss function (3.10) using stochastic gradient descent until convergence. We assign uniform measures $\pi \sim \text{Unif}(\Omega)$ to the interior
domain, and \(\pi^b \sim \text{Unif}(\partial \Omega)\) to the boundary domain. For the parameter space measure we compare two approaches. First is an adaptive surrogate computed using the data adaptive training scheme described in Section 3.2 and this is compared to a general surrogate trained over \(\pi^\alpha = \text{Unif}(A)\) where \(A\) is the general parameter space \(A = \bigcup_{i+j \leq 10} A_{i,j}\), where \(A_{i,j} = [-80/2^k, 80/2^k]\) for \(k = \max(i + j - 3, 0)\). This set is broad enough to cover all of the parameter values that we reasonably expect to infer, whilst also being prevented from becoming unnecessarily complex by the decay in the coefficients of higher degree terms.

**HMC calibration**

The surrogate is used to draw 20,000 MCMC samples from the posterior distribution. The first 10,000 of these constitute a warm-up period during which hyperparameters of the MCMC scheme are automatically determined. In this period the step size of a leap-frog integrator of the Hamiltonian dynamics is calibrated such that an acceptance rate of about 65\% is achieved. Furthermore the HMC mass matrix (as defined in the appendix) is updated such that it is proportional to the inverse of the empirical covariance of the existing sample during this period. Both of these choices are generally regarded as optimal for HMC. After the warm-up period these hyperparameters are fixed, and delayed-acceptance HMC is used with a Crank–Nicolson solver to draw a further 10,000 samples from the posterior distribution. As comparisons we also perform RWMH and MALA. Similarly to our HMC implementation we use the warm up period to calibrate their proposal distribution/preconditioning matrix, and adapt the step size to target the corresponding optimal acceptance rates of 23.4\% and 57.4\% respectively.

**4.2 Simulation study**

To validate our method we provide a study using simulated data. For this we consider the PDE (2.1) on the domain \((t, x) \in \Omega = [0, 3600] \times [0.3, 1]\) with \(c_0 = 35,000, c_1 = 1, c_2 = 1\), and initial and boundary conditions

\[
\begin{align*}
    u(0, x) &= x, & x &\in [0.3, 1] \\
    u(t, 0.3) &= 0.3, & x &\in [0.3, 1] \\
    u(t, 1) &= 1, & t &\in [0, 3600].
\end{align*}
\]

(4.5)

We produce noisy data from this PDE by solving the equation for instances of \(Bi(t, x)\) using a high resolution Crank-Nicolson solver and adding Gaussian white noise to data subsampled from the solution at 152 locations as illustrated in Figure 2.
Our aim in the Bayesian inverse problem is to sample from the posterior distribution of $\alpha$ conditional on this data. The instances of $Bi(t, x)$ that we illustrate are direct samples from a grid based Gaussian process discretisation, that is the true $Bi(t, x)$ are not polynomials in these examples. The ill-posedness of this problem implies that there exist a wide range of Biot numbers that achieve a good fit to the data (see Figure 3 for some examples of this variability for the data in Figure 2). It is therefore unreasonable to think that we should be able to infer $Bi(t, x)$ with a very high degree of certainty. Instead what we seek is a quantification of this uncertainty that uses our prior distribution to restricts our inference to physically reasonable results.
Figure 3: Top: An example of a $Bi(t, x)$ that is physically unrealistic but achieves a good fit to the data shown in Figure 2. Middle: At discrete times the spatial profiles of this $Bi(t, x)$ are shown in blue. The profiles of the true $Bi(t, x)$ are plotted in red. The green curves are alternate examples of $Bi(t, x)$ that achieve a similar fit. Bottom: Temporal residuals between the data and solution fitted using this $Bi(t, x)$, demonstrating that the unrealistic $Bi(t, x)$ produces a good fit.

To train the adaptive surrogate we begin by finding the Laplace estimate. As shown in Figure 4, this provides a reasonable starting point for inferring the Biot number, however at discrete times the spatial profile reveals that the true Biot number used to generate the data is often significantly outside the 95% credible interval. This behaviour is observed consistently with different instances of $Bi(t, x)$ and so we conclude that the Laplace approximation is over-certain for this problem.
This poor quantification of the uncertainty is not unexpected, as the Laplace approximation is a local estimate based only on the posterior curvature at the MAP estimate. In particular, Figure 4 justifies the requirement for MCMC methods to quantify the uncertainty more reliably. In our experiments we carry out RWMH, MALA, and HMC using the general and adaptive surrogates. After a warm up period each of these schemes is run for 10,000 iterations, both in a surrogate only setting, and with delayed-acceptance using Crank–Nicolson. In each case, Table 1 shows the accuracy of the surrogate, time taken in seconds for training and sampling, as well as the effective sample size and cost (measured as the time per effective sample). In this table the timings, ESS, and costs given are the averages over runs with 5 different datasets, and the $L^1$ errors correspond to the averaged difference between the surrogate and a high resolution Crank–Nicolson solver at the MAP parameter values.
### Table 1: Average timings, accuracy, and ESS achieved by each surrogate type after running 10,000 iterations of each sampling scheme.

| Surrogate type | Proposal | Surrogate only | Delayed-acceptance |
|----------------|----------|----------------|---------------------|
| Adaptive       | RWMH     | Time | ESS | Cost | Time | ESS | Cost |
| Training time: 925s | MALA     | 70.76 | 32.90 | 2.151 | 673.62 | 20.28 | 33.216 |
| L¹ error: 5.02 × 10⁻⁴ | HMC     | 110.84 | 427.34 | 0.259 | 799.51 | 364.97 | 2.191 |
| General        | RWMH     | 379.96 | 11263.20 | 0.034 | 1337.94 | 6497.04 | 0.206 |
| Training time: 15,874s | MALA   | 68.71 | 36.69 | 1.873 | 665.35 | 23.79 | 27.978 |
| L¹ error: 1.14 × 10⁻² | HMC     | 109.32 | 372.30 | 0.294 | 797.28 | 26.85 | 29.694 |

Beginning from a randomly initialised network, the time taken to train the general surrogate is 4 hours and 25 minutes, which is significantly longer than the 15 minutes taken to train the adaptive surrogate. In principle the general surrogate has the advantage of being applicable to multiple datasets once trained, however the larger error of this surrogate prevents it from being a reliable option. This is because proposals generated using the surrogate are almost always rejected by the secondary acceptance criteria in delayed-acceptance due to the discrepancy between the surrogate and the CN solution, furthermore Figure 8 shows that the inferences obtained by the general surrogate without delayed-acceptance are inaccurate. If available however, a general surrogate can significantly accelerate the training of an adaptive surrogate if used as the network’s initialisation. This acceleration can even apply if the general surrogate was trained to solve alternate variants of the PDE; for example it takes 171 seconds on average to train an adaptive surrogate to the same level of accuracy when initialised by a general surrogate trained to solve a version of the PDE defined for real experimental data (detailed in [7]). This alternative PDE has different coefficients \( \{c_0, c_1, c_2\} \) and non-linear boundary conditions determined by experimental data, yet despite these differences our approach constructs an accurate surrogate from this initial point in under 3 minutes.

Comparing sampling schemes we see that RWMH has the highest cost, followed by MALA. HMC significantly outperforms these methods despite taking longer to perform each iteration, obtaining an ESS that is slightly larger than the number of samples in the surrogate only setting. The introduction of the additional acceptance criteria in delayed-acceptance lowers the ESS since it reduces number of transitions that take place. Moreover, the introduction of delayed-acceptance decreases the average speed of each iteration due to the CN solver being executed. It is therefore notable that the cost of DA-HMC is still lower than...
surrogate only RWMH or MALA, highlighting that the optimisation of the proposal strategy is a significant factor that, if made efficient, can outweigh numerical solver inefficiencies when performing PDE-based MCMC sampling.

To investigate the accuracy of the posteriors achieved using each method we use delayed-acceptance HMC sampling as a baseline, as this is provably accurate and achieves a sufficiently large ESS. Figure 5 compares the credible intervals achieved by each proposal distribution in the surrogate only setting. Figure 6 compares the credible intervals achieved with and without delayed-acceptance using adaptive and general surrogates. Example trace plots for the coefficient $\alpha_3$, to the term that is cubic in space and constant in time are shown in Figure 7; these visualise the samples achieved by each delayed-acceptance scheme over a fixed runtime. Finally in Figure 8 for the same coefficient, we show how these differences in proposal distribution and surrogate choice affect the posterior accuracy by comparing the densities of their samples. These results show that in our experiments the adaptive surrogate HMC approach without delayed-acceptance achieves results that are very close to the true posterior with delayed-acceptance, albeit this behaviour is not mathematically guaranteed a-priori. All other approaches perform poorly in comparison. For RWMH and MALA with an adaptive surrogate this is due to their lower ESS. For the general surrogate the lower accuracy results in a sampling bias in the surrogate only setting, resulting in $Bi(t, x)$ being underestimated in comparison to the true posterior. Moreover for the general surrogate, when coupled with delayed-acceptance it exhibits an extremely high rejection rate resulting in a poor Monte Carlo estimate of the posterior.
Figure 5: Spatial profiles of the 95% credible intervals of surrogate only adaptive HMC, MALA, and RWMH at various time points. The red curves are the spatial profiles of the true Biot number used to simulate the data.
Figure 6: Spatial profiles of the 95% credible intervals of surrogate only adaptive HMC, delayed-acceptance adaptive HMC, surrogate only general HMC, delayed-acceptance general HMC. The red curves are the spatial profiles of the true Biot number used to simulate the data.
Figure 7: Comparison of trace plots of delayed-acceptance HMC, MALA, and RWMH and their 50 step moving average. The lengths of these chains is chosen so that the time taken to achieve each sample is the same.

Figure 8: Comparison of sample densities of different schemes for the Chebyshev coefficient that is cubic in space and constant in time.

In future we will apply this methodology in a range of more complex experimental settings,
such as experiments with variable frequencies that mimic the engine activity at various stages of a flight.

5 Closing remarks

In this work we have developed a fully Bayesian approach to solving the PDE inverse problem for the unknown spatio-temporal Biot number. Our approach augments the general deep surrogate method for parametric PDEs by implementing a novel training scheme based on solving the PDE only over the approximated posterior distribution. To achieve this, we have described how to apply our deep learning approach to quickly obtain a Laplace estimate. By comparing this to the true posterior we have demonstrated that the Laplace approximation gives reasonable estimates of the Biot number, though it underestimates the overall uncertainty of this estimate. To sample from the posterior in an accurate and efficient manner, we applied a novel deep surrogate-based delayed-acceptance HMC scheme. This scheme utilises the fast evaluation and differentiation of the deep learning surrogate to make decorrelated proposals based on HMC trajectories and applies delayed-acceptance acceptance criteria to mathematically guarantee the accuracy of our posterior with respect to a Crank–Nicolson solver.

Since Biot number calculations are involved in the design of turbo-machinery, it is important that the results of our inferences are have quantifiable accuracy. The delayed-acceptance step ensures sampling accuracy is consistent with a Crank–Nicolson solver, and our results demonstrate that the delayed-acceptance HMC scheme efficiently achieves a large effective sample size, thus ensuring a low statistical error. Our scheme achieves this accuracy while maintaining a lower cost per effective sample than Metropolis–Hastings schemes based on random walk or Langevin proposals, even if these schemes sacrifice the sampling accuracy of the delayed-acceptance step and rely solely on rapid surrogate evaluations. We also showed that the higher accuracy of our adaptively trained surrogate combined with HMC resulted in accurate posterior samples at a fraction of the cost. Our methodology is generalisable to other PDEs and applications, therefore in settings where approximate accuracy is sufficient the potential exists to apply this approach for substantial computational gains.
Appendix

MAP estimation using local deep surrogates

To train the Laplace surrogate we must first compute the MAP estimate for the posterior in (3.7), that is we are interested in finding

\[
(\alpha^*, \sigma^*) = \text{argmax} \left\{ \log p(\alpha, \sigma | \hat{t}, \hat{x}, \hat{z}) \right\}.
\]  

Starting with an initial guess \((\alpha_0, \sigma_0)\), we solve this optimisation problem using an approach that is similar in concept to a trust region method. More specifically, we train a local surrogate \(\hat{u}(t, x, \alpha)\) at \(\alpha_0\) by minimising the loss function (3.10) with \(\pi^0 = \pi^{\alpha_0}\), where \(\pi^{\alpha_0}\) is a probability measure with mass focused locally around \(\alpha_0\). When the local surrogate is accurate, as judged by achieving a sufficiently low loss function value, we substitute this into the log-posterior (5.1) and take a gradient ascent step with respect to \((\alpha_0, \sigma_0)\) to maximise the resulting expression. This process repeats in an alternating manner, whereby at iteration \(n\) the network is trained over \(\pi^{\alpha_n}\) to ensure local accuracy at \(\alpha_n\) and another gradient ascent step is performed to maximise the posterior. Upon convergence the MAP estimate is the final value of \(\alpha_n\), and the Hessian matrix used in the Laplace approximation covariance is constructed by automatic differentiation of the log-posterior at \(\alpha_n\).

In our implementation we choose \(\pi^{\alpha_n} \sim \text{MVN}(\alpha_n, \Lambda^n)\) for the local measure, where \(\Lambda\) is a diagonal matrix. For the variance of this distribution we choose \(\Lambda^n_{i,i} = \lambda_n/2^k\), where \(k\) is the degree of the polynomial term corresponding to the \(i^{th}\) position. The decay in the variance of \(\pi^{\alpha_n}\) is manually set, starting with \(\lambda_0 = 20\) and ending with \(\lambda_{\text{end}} = 0.5\). This local surrogate is significantly cheaper to compute than the general surrogate, taking only a few seconds to reduce the loss function to below the level achieved by training a general surrogate for 15 hours, and ultimately achieving a significantly higher accuracy in a fraction of the time.

Hamiltonian Monte Carlo

Hamiltonian Monte Carlo \([4]\) is an MCMC approach, which uses the derivatives of the density function being sampled to generate efficient Metropolis–Hastings transitions spanning the posterior density \([4]\). A benefit to using HMC proposals over simpler proposals methods such as those based on Gaussian symmetric random walks is that HMC significantly reduces the correlation between successive sampled states while maintaining a high acceptance rate, meaning that far fewer MCMC iterations are needed to approximate the target probability distribution to a given tolerance level. This sampling efficiency is achieved by simulating random trajectories of a carefully designed Hamiltonian system to propose successive states,
whilst maintaining detailed balance with respect to the target distribution. Another benefit of HMC is short burn-in periods, since these Hamiltonian proposals target regions of higher probability. [3, 13] have noted that the HMC approach is particularly beneficial in Bayesian problems.

Consider the target distribution $\psi(\theta)$ and suppose we want to simulate a chain of samples $\theta_0, \theta_1, \theta_2, \ldots$ from this distribution. In HMC, we expand the target density into a 2D-dimensional distribution $\psi(\theta, p) = \psi(p|\theta)\psi(\theta)$ over a phase space described in terms of a position variable in the parameter space $\theta \in \Theta$ and an auxiliary momentum variable $p \in \mathbb{R}^M$. We then use Hamiltonian dynamics to sample from this density, noting that the target distribution is the marginal $\psi(\theta) = \int_{\mathbb{R}^M} \psi(\theta, p)dp$ which can be estimated from the joint sample by simply discarding the sampled momentums. Hamilton’s equations are given by the coupled ordinary differential equations

$$\frac{d\theta_i}{dt} = \frac{\partial H}{\partial p_i},$$

$$\frac{dp_i}{dt} = -\frac{\partial H}{\partial \theta_i}.$$  (5.2)

Here, $\theta_i$ and $p_i$ represent the $i$th component of the position and momentum vectors, and $H$ is the Hamiltonian. For more details on Hamiltonian systems, see, for example [11]. In HMC, we define the Hamiltonian by decomposing the phase space distribution

$$H(\theta, p) = -\log \psi(p, \theta)$$

$$= -\log \psi(\theta) - \log \psi(p|\theta)$$

$$= U(\theta) + K(p|\theta)$$  (5.3)

Here $U(\theta) = -\log \psi(\theta)$ represents the potential energy and $K(p|\theta) = -\log \psi(p|\theta)$ represents the kinetic energy of the system. For our application we define the conditional momentum distribution as $\psi(p|\theta) \sim \text{MVN}(0, M)$ for some symmetric positive definite matrix $M$, then the Hamiltonian takes the form

$$H(\theta, p) = -\ln(\psi(\theta)) + \frac{1}{2}p^TM^{-1}p.$$  (5.4)

At the beginning of each Monte Carlo iteration the initial momentum is sampled from $\psi(p|\theta)$. Using this initial momentum the Hamiltonian dynamics are simulated starting at the current state of the position variable of the Markov Chain. These simulations then follow contours of constant energy in phase space over a pre-defined time horizon. The matrix $M$ (known as the mass matrix due to its physical interpretation) determines the initial speed and directions of the trajectories, and therefore has a large impact on the proposals. We specify this matrix adaptively within a warm-up phase prior to sampling as outlined in
In order to use the Hamiltonian dynamics within the proposal distribution for a Markov Chain we require a mechanism for simulating these trajectories. This is achieved using the leapfrog integrator, which conserves the energy in the system, leading to accurate simulations relative to non-symplectic solvers. Given a step size $\delta$, a single leapfrog iteration of the system (5.2) is given by the steps:

\[
\begin{align*}
    p_i(t + \delta/2) &= p_i(t) - (\delta/2) \frac{\partial U}{\partial \theta_i(t)}, \\
    \theta_i(t + \delta) &= \theta_i(t) + \delta \frac{\partial K}{\partial p_i(t + \delta/2)}, \\
    p_i(t + \delta) &= p_i(t + \delta/2) - (\delta/2) \frac{\partial U}{\partial \theta_i(t + \delta)}.
\end{align*}
\]  

(5.5)

Multiple of these steps can be carried out in order to integrate over longer time horizons, and different step sizes can be used depending on the accuracy required. The state of the position and momentum variables at the end of the simulation are used as proposed state variables $\theta_{\text{prop}}$ and $p_{\text{prop}}$. Finally the Metropolis-Hastings acceptance criteria is employed (on the phase space distribution) to ensure detailed balance is maintained. The acceptance probability in this case is

\[
    A_H(\theta_{\text{prop}}, p_{\text{prop}}, \theta, p) = \min \left\{ 1, \frac{\psi(p_{\text{prop}}|\theta_{\text{prop}})\psi(\theta_{\text{prop}})}{\psi(p|\theta)\psi(\theta)} \right\}.
\]  

(5.6)

Combining these steps, the HMC algorithm for drawing $N_{\text{samples}}$ samples from a target distribution can be written as:

**Algorithm 2: Hamiltonian Monte Carlo**

1. Set initial position state $\theta_0 \in \Theta$.
2. for $i = 0, 1, \ldots, N_{\text{samples}}$ do
   1. Sample momentum $p_i \sim MVN(0, M)$.
   2. Simulate Hamiltonian with initial state $(\theta_i, p_i)$ using leapfrog algorithm to obtain proposed states $(\theta_{\text{prop}}, p_{\text{prop}})$.
   3. Sample $u \sim U(0, 1)$;
   4. if $u \leq A_H(\theta_{\text{prop}}, p_{\text{prop}}, \theta, p)$; then
      1. set $\theta_{i+1} := \theta_{\text{prop}}$ (accept);
   5. else
      1. set $\theta_{i+1} := \theta_i$ (reject);
6. end
3. return $\theta = (\theta_0, \theta_1, \ldots, \theta_{N_{\text{samples}}})$
Acknowledgements

We thank our colleague Hui Tang from the Department of Mechanical Engineering at the University of Bath for their helpful discussions regarding the context of this work.

Teo Deveney is supported by a scholarship from the EPSRC Centre for Doctoral Training in Statistical Applied Mathematics at Bath (SAMBa), under the project EP/L015684/1.

References

[1] M. Abadi et al. Tensorflow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org, 2015.

[2] A. Alexiou, N. J. Hills, C. A. Long, A. B. Turner, and J. A. Millward. Heat transfer in high-pressure compressor gas turbine internal air systems: A rotating disc-cone cavity with axial throughflow. *Exp. Heat Trans.*, 13(4), 2000.

[3] Michael Betancourt. A conceptual introduction to Hamiltonian Monte Carlo. *arXiv preprint arXiv:1701.02434*, 2017.

[4] Michael Betancourt and Mark Girolami. Hamiltonian Monte Carlo for hierarchical models. *Current trends in Bayesian methodology with applications*, 79:30, 2015.

[5] J. Christen and Colin Fox. Markov chain Monte Carlo using an approximation. *Journal of Computational and Graphical Statistics - J COMPUT GRAPH STAT*, 14:795–810, 12 2005.

[6] T. Deveney, E. Mueller, and T. Shardlow. A deep surrogate approach to efficient Bayesian inversion in PDE and integral equation models, 2019. arXiv:1910.01547.

[7] T. Deveney, T. Shardlow, G. D. Lock, and H. Tang. Use of bayesian statistics to calculate transient heat fluxes on compressor discs. *In preparation. To be submitted to Physics of Fluids.*, 2022.

[8] Y. Efendiev, T. Hou, and W. Luo. Preconditioning Markov chain Monte Carlo simulations using coarse-scale models. *SIAM J. Scientific Computing*, 28:776–803, 01 2006.

[9] Charles J. Geyer. Practical Markov Chain Monte Carlo. *Statistical Science*, 7(4):473–483, 1992.

[10] Gitta Kutyniok, Philipp Petersen, Mones Raslan, and Reinhold Schneider. A theoretical analysis of deep neural networks and parametric PDEs. *arXiv preprint arXiv:1904.00377*, 2019.
[11] L.D. Landau, E.M. Lifshitz, J.B. Sykes, and J.S. Bell. Mechanics. Butterworth-Heinemann. Elsevier Science, 1976.

[12] G J Lord, C E Powell, and T Shardlow. An Introduction to Computational Stochastic PDEs. Cambridge University Press, 2014.

[13] R. M. Neal. Bayesian Learning for Neural Networks, volume 118. Springer Science & Business Media, 2012.

[14] Michael Renardy and Robert C Rogers. An introduction to partial differential equations, volume 13 of Texts in Applied Mathematics. Springer Science & Business Media, New York, 2 edition, 2006.

[15] J. Sirignano and K. Spiliopoulos. Dgm: A deep learning algorithm for solving partial differential equations. Journal of Computational Physics, 375:1339 – 1364, 2018.

[16] A. M. Stuart. Inverse problems: A Bayesian perspective. Acta Numerica, 19:451–559, 2010.

[17] H. Tang, T. Shardlow, and J. M. Owen. Use of Fin equation to calculate Nusselt numbers for rotating discs. Journal of Turbomachinery, 12(137), 2015.

[18] J W Thomas. Numerical Partial Differential Equations: Conservation Laws and Elliptic Equations. Numerical Partial Differential Equations. Springer, 1999.

[19] L. N. Trefethen. Approximation Theory and Approximation Practice. SIAM, 2013.

[20] Liu Yang, Xuhui Meng, and George Karniadakis. B-pins: Bayesian physics-informed neural networks for forward and inverse PDE problems with noisy data. Journal of Computational Physics, 425:109913, 01 2021.

[21] Yinhao Zhu, Nicholas Zabaras, P. Koutsourelakis, and Paris Perdikaris. Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data. Journal of Computational Physics, 394, 05 2019.