Received: 19 March 2020 | Revised: 7 November 2020 | Accepted: 18 November 2020

DOI: 10.1002/cnm.3421

RESEARCH ARTICLE - FUNDAMENTAL

WILEY

Markov chain Monte Carlo with Gaussian processes for fast parameter estimation and uncertainty quantification in a 1D fluid-dynamics model of the pulmonary circulation

L. Mihaela Paun | Dirk Husmeier

School of Mathematics and Statistics, University of Glasgow, Glasgow, UK

Correspondence
L. Mihaela Paun, School of Mathematics and Statistics, University of Glasgow, Glasgow G12 8QQ, UK.

Funding information
Engineering and Physical Sciences Research Council, Grant/Award Number: EP/N014642/1

Abstract

The past few decades have witnessed an explosive synergy between physics and the life sciences. In particular, physical modelling in medicine and physiology is a topical research area. The present work focuses on parameter inference and uncertainty quantification in a 1D fluid-dynamics model for quantitative physiology: the pulmonary blood circulation. The practical challenge is the estimation of the patient-specific biophysical model parameters, which cannot be measured directly. In principle this can be achieved based on a comparison between measured and predicted data. However, predicting data requires solving a system of partial differential equations (PDEs), which usually have no closed-form solution, and repeated numerical integrations as part of an adaptive estimation procedure are computationally expensive. In the present article, we demonstrate how fast parameter estimation combined with sound uncertainty quantification can be achieved by a combination of statistical emulation and Markov chain Monte Carlo (MCMC) sampling. We compare a range of state-of-the-art MCMC algorithms and emulation strategies, and assess their performance in terms of their accuracy and computational efficiency. The long-term goal is to develop a method for reliable disease prognostication in real time, and our work is an important step towards an automatic clinical decision support system.

KEYWORDS

classification, emulation, Gaussian processes, MCMC, pulmonary circulation, uncertainty quantification

1 | INTRODUCTION

Over the past few decades, cardiovascular modelling has seen an impressive advancement in terms of enabling better understanding of cardiovascular (patho)physiology, as well as assisting clinicians in the diagnosis, prognostication and treatment of cardiovascular diseases.[1] Moreover, cardiovascular modelling can aid in surgical interventions, in the design and evaluation of medical devices, and even in the inference of unknown and immeasurable parameters from measured data.[1] Before being used as a decision support system in the clinic, these models must first be adapted to patient-specific conditions and an assessment of their credibility (uncertainty quantification, UQ) based on a comparison between model predictions and clinical data, must be performed. For example, studies in References 1 and 2 discuss the requirements that clinically applicable cardiovascular models must meet and the advances required to do so. Parameter inference and uncertainty quantification in biophysical models of physiological processes, and cardiovascular processes in particular is a challenging task to accomplish, as the physical models, typically expressed in terms of coupled non-linear PDEs, with no closed-form solution, are becoming more complex. This leads to a growing number of unknown biophysical parameters, which typically cannot be directly measured non-invasively and hence have to be inferred from limited, potentially noisy clinical data, for which the acquisition process may be costly and require clinical personnel with expertise.

In the cardiovascular mathematical modelling community, most studies addressing parameter estimation focus on frequentist approaches with parameter optimisation, where the standard approach is to search for parameter configurations that lead to a good agreement between the clinical data and the predictions from the physical model by a minimisation of an objective function that quantifies the mismatch between predictions and data (typically the Euclidean distance).[3-6] However, an approach solely focused on parameter optimisation does not provide a direct mechanism for UQ. In contrast, sampling-based Bayesian methods facilitate the uncertainty analysis via the exploration of the posterior distribution of the parameters given the available data, and can deal with the often multi-modal and high-dimensional posterior distributions of the parameters. However, sampling from the posterior distribution requires repeatedly calculating the likelihood by numerically integrating the PDEs, which renders the sampling process slow, and thus unattractive for clinical practitioners. In addition, finding efficient algorithms that return high effective sample sizes in a reasonable time frame is challenging, especially when there are strong posterior correlations among the parameters. Our study employs Bayesian methods and demonstrates how the issues outlined above can be overcome in practice, and how the parameter estimation and UQ can be made tractable.

The particular application of our study is a 1D fluid-dynamics model of the pulmonary circulation,[6] which predicts blood flow and pressure in the lungs by solving a system of non-linear PDEs. The haemodynamic predictions, computed in an arterial network model constructed from micro-computed tomography (CT) images from a control mouse, are compared to dynamic pressure data in the main pulmonary artery (MPA). The 1D model[6] is characterised by two types of parameters: specifying the vessel network (i.e., vessel geometry or connectivity of the arteries), and the haemodynamics (i.e., pulmonary blood pressure and flow). The focus of this study is the inference of parameters describing the haemodynamic equations in a fixed vessel network obtained from one image segmentation. The parameters estimated are the vessel wall stiffness, which can only be measured ex-vivo, and parameters specifying the PDE boundary conditions at each terminal vessel (resistance and capacitance of Windkessel models), that is, a total of four parameters.

Recently, a few cardiovascular modelling studies,[7-9] which employ Bayesian (probabilistic) approaches, have emerged, but a lot of work still needs to be done in this field. In our previous work,[7] we have shown how parameters of a system of PDEs describing the pulmonary circulation can be inferred using MCMC, the Delayed Rejection Adaptive Metropolis (DRAM) algorithm.[10] This is referred to as the Bayesian analysis of computer code models, as the computer code defines a deterministic system that takes certain parameter values as input and outputs the data – for an overview of Bayesian calibration of computer models, see.[11-14] Similarly, Tran et al.[8] employ an adaptive MCMC algorithm using differential evolution with self-adaptive randomised subspace sampling (DREAM) for the coronary circulation. For a review on uncertainty and variability in computational and mathematical models of cardiac physiology, the reader is referred to.[15] In addition,[16] reviews the application of Bayesian methods to bioinformatics (e.g., protein informatics) and computational systems biology (e.g., quantitative network models). We also mention[17,18] for an application of MCMC to mathematical models for tumour growth. Although rare, in cardiovascular modelling, examples of studies employing MCMC to sample from multi-modal distributions are[19]: with an application in soft tissue mechanics – constitutive modelling of tendon ligaments in mice, and[20] with an application in haemodynamic models – modelling of pulmonary hypertension with a lumped parameter model on human data.

The MCMC algorithms can either be based on a random walk (RW, e.g., Metropolis-Hastings[21]), or can introduce auxiliary variables for exploitation of gradient information from the posterior distribution (Hamiltonian Monte Carlo - HMC[22]). All studies employing Bayesian methods in the cardiovascular modelling community use RW MCMC algorithms. However, the HMC algorithm has been shown to outperform the RW algorithms in terms of efficiency (e.g., for a 100-dimensional multivariate Gaussian distribution as a target density, see Chap. 5 in References 23 or 24). Our study is the first to employ HMC-type algorithms in cardio-physiological modelling.

More generally, HMC has rarely been applied to non-linear ODE or PDE models. Four noticeable exceptions are,[25] where the authors applied HMC to infer the parameters of an ODE model of intracellular processes,[26] where HMC is applied to a PDE-based model of tumour growth, and,[27] where an HMC extension (Riemann Manifold HMC) is applied to a PDE model of steady state heat conduction, or,[24] where Hamiltonian Monte Carlo algorithms are applied to a set of ODEs describing dynamic causal models.

The principal disadvantage of applying HMC to inference in PDE models is the high computational cost associated with numerically integrating the PDEs a large number of times. HMC trajectories are simulated by following a set of deterministic Hamiltonian dynamics steps in parameter space. Throughout each trajectory, the PDEs are evaluated multiple times (for calculating the likelihood and its gradient) until a proposal is made, unlike the RW algorithms, which require one single PDE evaluation for a proposal to be made. For the calculation of the likelihood gradient, three approaches may be taken: finite differences,* forward sensitivities† (see the study in Reference 28 for a computation of sensitivity derivatives of Navier–Stokes equations in computational fluid-dynamics), and the adjoint method‡.[24,29] For example, Sengupta et al.[24] compare the performance in terms of computational speed and accuracy of the three methods on an ODE system, and the adjoint method was shown to be superior (specific details of these methods can also be found in Reference 29).

Regardless of the method chosen, calculating the likelihood gradients is very computationally intensive if performed repeatedly within the HMC algorithm. To address this shortcoming, two approaches have been proposed in the literature. One approach is a stochastic-gradient HMC, introduced in Reference 30, and the main idea is to subsample the data, which introduces noise, and thus the full-data gradients in the Hamiltonian equations are replaced by stochastic gradients, which can result in reduced exploration efficiency and accuracy.[31] Another approach to speed up HMC is the use of computationally cheaper surrogate models for the expensive likelihood (or posterior distribution).[32,33] In Appendix A.1 we give a brief review of surrogate modelling.

HMC can be coupled with the surrogate model,[34] and two main approaches can be taken to accomplish this: one which draws samples from the approximate posterior distribution defined by the surrogate model, and a second one which draws samples from the asymptotically exact posterior distribution. The first approach, taken by most studies on cardiac models,[35-40] and in the broader literature,[41-47] samples from the posterior distribution induced by the surrogate model, which was constructed and refined prior to the sampling. While this leads to substantial gains in computational efficiency, it introduces a bias, that is, efficiency is gained at the expense of accuracy, as the inference relies heavily on the quality of the approximation model. Cotter et al.[48] theoretically show that the bias could in principle be bounded, but no practical suggestions to do so are offered. Schiavazzi et al.[36] verify that the bias introduced is small enough to be negligible in a surgery haemodynamics model for single ventricle palliation. This is accomplished by a forward UQ based on the parameter posterior samples obtained from the surrogate posterior distribution. The forward UQ allows to determine if the variability bounds on the model predictions are compatible with the expected measurement error. If the bias introduced by the emulator is too large, the model predictions may have unrealistically narrow variability bounds (bias-variance trade-off). This however relies on knowing the variance of the measurement error, which may not always be the case. To combat this issue, exact methods, in the sense of asymptotic convergence to the exact posterior distribution and can be used. One such method corrects for the bias by using the surrogate for the proposal only, for example, a study in Reference 32 couples HMC with a surrogate built using Gaussian Processes (GPs). Another study[33] couples HMC with a surrogate created using random bases, that can scale better with high-dimensional spaces than GPs. The latter study also introduces a method which refines the surrogate continually as MCMC proceeds (and thus reduces the number of training points required), while the asymptotic exactness is ensured. We also mention,[49-52] in which a similar approach is taken. For example, the approach in Reference 50 uses forward simulations from the PDEs for incrementally refining a local approximation of the (unnormalised) log posterior. However, while the authors prove the asymptotic consistency of their method, it depends on various heuristic parameters, which critically affect the computational efficiency and may be difficult to tune in practice. The Delayed Acceptance (DA)[53,54] scheme is another exact method, and its advantage is that it does not depend on any heuristically set terms, which motivates us to utilise it in our study. This method, employed in References 53-57, is a two-stage acceptance procedure, with two separate acceptance/rejection decisions. The idea is that the first decision is a computationally fast pre-filter step, which upon rejection of a proposed new parameter avoids having to carry out the computationally expensive second step.

In our work, we follow[32,58] and combine HMC§ with statistical emulation using GPs, a probabilistic inference method used to construct a surrogate model (known as the "emulator") for the (unnormalised) log posterior distribution, and we implement this in a DA framework. Throughout the trajectory, the HMC algorithm is run at low computational costs on this surrogate function, and the Metropolis-Hastings acceptance/rejection step uses the ratio of the true posterior distributions. This only requires the PDEs to be numerically integrated once throughout the HMC trajectory, and thus substantially reduces the computational complexity. The algorithm is exact in the sense of converging to the true posterior distribution, assuming no discretisation errors are introduced from the numerical integration of the PDEs (investigation of the discretisation errors is beyond the scope of this paper). Among cardiovascular mathematical modelling studies, we have found a single study in cardiac electrophysiology which uses a similar idea to ours: Dhamala et al.[61] employ the Metropolis-Hastings algorithm coupled with GPs and DA, and they take inspiration from a study by Lê et al.,[26] who apply the HMC algorithm coupled with GPs to a tumour growth model. Both of these studies build a surrogate model for the log posterior distribution, and the latter also illustrates how an MCMC algorithm run on a surrogate model with no bias correction step can produce inaccurate inference results. However, these studies apply one single algorithm to the problem of interest, without any comparative evaluation of its performance.

Our first main contribution is to perform an extensive comparison of algorithms in terms of their efficiency and accuracy, and conclude with an algorithm recommendation. In addition, our inference procedure deals with unknown constraints in parameter space, caused by the violation of the physical model assumptions. In such problems, naively employing a GP-MCMC method will yield inaccurate results, hence our second main contribution is the use of a multivariate classifier which automatically learns the infeasible parameter domain, and to our best knowledge, this is novel in the cardiovascular and mathematical biology research community. We therefore combine emulation with a series of state-of-the-art sampling methods which converge to the true posterior distribution and are particularly adapted to complex models (i.e., models that are extensive to sample from). Since time is a critical factor for clinical decision making, we place a strong focus on computational efficiency. Thus, in our extensive benchmark study we look for the algorithm that provides the best trade-off between accuracy and computational efficiency.

Our study presents the following methodological innovations. Firstly, we adapt the combination of GP emulation and HMC (henceforth referred to as the GPHMC algorithm), as proposed in Reference 32, to biophysical parameter estimation in a complex cardiovascular model describing the pulmonary circulation. We additionally include a GP classifier to deal with the a priori unknown regions in parameter space where the physical assumptions of the mathematical model are violated and the cardiovascular simulation software does not provide valid outputs. Secondly, we introduce the DA-GPHMC algorithm, which couples the DA algorithm based on Sherlock's work[55] with Rasmussen's GPHMC algorithm.[32] Thirdly, we extend this framework to algorithms which advance HMC: No U-turn sampler,[62] Riemann Manifold HMC,[63] and Lagrangian Dynamical Monte Carlo,[64] and we automatically adapt the algorithms' tuning parameters with Bayesian optimisation, following an idea by Freitas in Reference 65. Fourthly, we perform a novel comparative evaluation of these algorithms in the application context of mathematical modelling of cardiovascular models.

The paper is structured as follows. Section 2 introduces the mathematical cardiovascular model and describes the data. Section 3 revises some background on statistical methodology, related to GPs and HMC. Section 4 discusses the new statistical methodology for the present work. Section 5 provides an overview of the setup of our simulation study. Section 6 presents our numerical results. Finally, Section 7 concludes with a discussion and an outlook on future work.

2 | APPLICATION TO THE PULMONARY BLOOD CIRCULATION

This section describes the 1D fluid-dynamics model utilised in this study, as well as the physiological data and synthetic data on which our inference procedure is carried out.

2.1 | Fluid-dynamics model

The particular focus of our work is a fluid-dynamics model of the pulmonary blood circulation. This is part of a larger research project aiming to address the challenging task of non-invasively diagnosing long-term hypertension (high blood pressure) in the pulmonary system, that is, the blood vessel network connected to the right ventricle of the heart. Long-term hypertension is a major risk factor for a variety of medical conditions, including coronary artery disease, stroke and heart failure. Non-invasive measurement of the blood pressure in the systemic circuit, that is, the blood vessel network connected to the left ventricle of the heart, is straightforward; one just has to use a sphygmomanometer. However, sphygmomanometers cannot be applied to the pulmonary circuit. State-of-the-art medical procedures are based on right-heart catheterisation, which is an indispensable tool, providing important information about the function and structure of the measured physiological quantities. Catheterisation is typically performed once the symptoms of the disease have already appeared, and almost never used for screening and early diagnosis. Model-based diagnostics used in the early stages of the disease could avoid a patient developing symptoms, and thus avoid the risks and possible side effects of right-heart catheterisation (e.g., excessive bleeding because of puncture of the vein during catheter insertion, and partial collapse of the lung). The ultimate quest, therefore, is to combine magnetic resonance imaging (MRI) with mathematical modelling and statistical inference to be used as a disease diagnosis tool, likely with most impact in the early disease stages.

The present work is a first stepping stone in this direction. We use a 1D fluid-dynamics model developed by the authors of.[6,66] Compared to 3D fluid-dynamics models,[67,68] 1D models can predict nearly identical pressure and flow waves throughout the pulmonary network at a fraction of the computational cost, making them ideal as a real-time clinical tool. Therefore, the 1D models are often used as computationally tractable approximations to 3D models.[38] We use measured blood flow and pressure time series in the MPA to infer various biophysical parameters, including the vessel stiffness, which could provide important indicators for pulmonary hypertension. Subsequent studies will investigate this association more closely. The focus of the present study is the assessment of the accuracy and computational efficiency of the statistical inference and uncertainty quantification procedure.

We apply the computational inference procedure discussed in the next section, with different MCMC variants, as reviewed in Section 3.2 to a 21-vessel network model[6,66], which predicts 1D pulmonary arterial flow and pressure by solving a system of PDEs (see Figure 1 for a schematic of the arterial network of a healthy mouse lung). The model is derived from the incompressible axisymmetric Navier–Stokes equations for a Newtonian fluid, see Equation (1). The system of equations is closed with a constitutive equation describing the elasticity (i.e., stiffness) of the blood vessel walls, see Equation (2):

$$\frac{\partial A}{\partial t}+\frac{\partial q}{\partial x}=0,\quad \frac{\partial q}{\partial t}+\frac{\partial}{\partial x}\frac{q^2}{A}+\frac{A\partial p}{\rho\partial x}=-\frac{2\pi\mu r}{\delta}\frac{q}{A}, \tag{1}$$

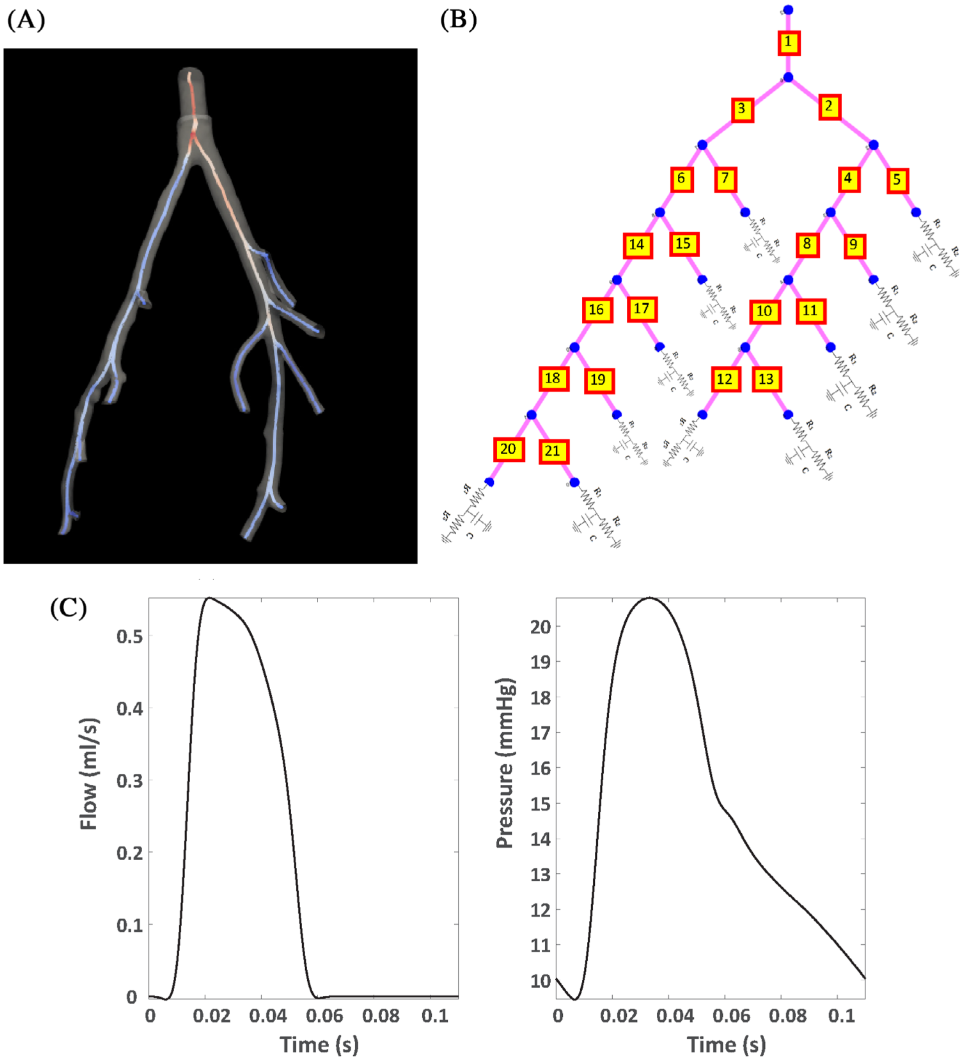

FIGURE 1 Panel (A): 3D smoothed network from a micro-CT image of a healthy mouse lung. Panel (B): the connectivity graph of the same network. A three-element Windkessel model with two resistors and a capacitor is attached at the end of each terminal vessel. Panel (C): Measured pulmonary blood flow (left) and pressure (right) in the main pulmonary artery of a healthy mouse. These data resulted from averaging of measurements across 20 heart beats (raw data are not available). The flow data are used as inlet boundary conditions in the PDEs (1)–(2), while the pressure data are used in the statistical inference procedure. Figures in panels (A) and (B) are taken from our study in 7

$$p = p_0 + \frac{4}{3}s\left(1 - \sqrt{\frac{A_0}{A}}\right), \tag{2}$$

where x(cm) and t(s) are the longitudinal spatial and temporal coordinates, p(mmHg) is the blood pressure, q(ml/s) is the blood flow rate, A(cm^2) is the cross-sectional area, A_0 is the unstressed vessel cross-sectional area, p_0(mmHg) is the transmural pressure, s(mmHg) is the arterial network stiffness, $\rho = 1.055$g/ml is the blood density, $\mu = 0.049$ g/(cm s) is the viscosity and $\delta = \sqrt{\mu T/2\pi\rho}$cm is the boundary-layer thickness of the velocity profile. The boundary conditions are specified as follows. As inlet boundary conditions, an inlet flow at the MPA is prescribed, which is obtained from haemodynamic data. At the vessel junctions, conservation of flow ($q_\mathrm{p} = q_{\mathrm{d}_1} + q_{\mathrm{d}_2}$) and continuity of pressure ($p_\mathrm{p} = p_{\mathrm{d}_1} = p_{\mathrm{d}_2}$) are ensured; here p represents the parent vessel, and d_1 and d_2 represent the daughter vessels. As outflow boundary conditions, 3-element Windkessel models (two resistors R_1 and R_2, and a capacitor, C) are used; they are attached at every terminal artery, and are of the form

$$Z(\omega) = R_1 + \frac{R_2}{1 + i\omega C R_2} \Rightarrow q(L,t) = \frac{1}{T}\int_0^T p(L, t-\tau) Z(\tau) d\tau, \tag{3}$$

where $Z(\omega)$ is the impedance, ω is the angular frequency, T is the length of the cardiac cycle, R_1, R_2(mmHg s/ml) are the two resistances, and C(ml/mmHg) is the capacitance.

2.2 | Model parameters

We follow[6] and assume that the vessel stiffness s, is constant across the network – subsequent studies will relax this assumption. The three Windkessel elements (R_1, R_2, C) vary across the different terminal arteries. First, their nominal values are calculated for every terminal vessel j ($R_{01}^j, R_{02}^j, C_0^j$ in Equation (4)),[7] then scaling factors r_1, r_2, c for these estimates are introduced, and they are kept constant across all terminal arteries:

$$R_1^j = (1 - 0.5r_1)R_{01}^j, \quad R_2^j = (1 - 0.5r_2)R_{02}^j \quad \text{and} \quad C^j = (1 - 0.5c)C_0^j. \tag{4}$$

Thus, each outlet has different nominal values for the resistances and the capacitor, and all outlets are subsequently adjusted in the same manner.

The vector of parameters to be estimated is 4-dimensional,

$$\boldsymbol{\theta} = (s, r_1, r_2, c). \tag{5}$$

These parameters lie within biologically plausible ranges (see Table 1), as established by the authors of[6] based on the literature[69] and the authors' experience running the mathematical solver. However, these ranges are univariate, and the parameters' behaviour in the joint space is not known prior to carrying out the analysis. Thus, we may need to deal with the fact that for certain parameter values and combinations, the underlying physical assumptions of the model are violated. Further details on this are given in Section 4.

Parameter	Range
s	$[7 \times 10^4, 5 \times 10^5]$
r_1	$[-0.5, 1.92]$
r_2	$[-0.5, 1.0]$
c	$[-2.5, 1.5]$

Note: Here s is the vessel stiffness parameter (expressed in g/cm/s^2 units) and r_1, r_2, c are the Windkessel adjustment parameters (dimensionless).

TABLE 1 Univariate (prior) parameter ranges for the PDE parameters of the fluid-dynamics model described in Section 2

2.3 | Physiological data

The data for our study come from Qureshi et al.[6] These are blood flow and pressure measurements in the arterial vessel network of a healthy mouse lung (see Figure 1). A summary of the experimental protocols used to extract the haemodynamic and image data can be found in References 6 and 66. For more detailed experimental protocols, the reader is referred to.[70,71] During heart contraction, blood is pumped into the arterial vessels. Waves of blood flow and pressure get propagated along the arterial circulation due to the pulsatile rhythm of the heart and the elasticity of the vessel wall. The blood flow is measured non-invasively with ultrasound, and the pressure is measured invasively with right-heart catheterisation.[70] The raw data consisted of several time series of pulmonary blood pressure and flow measurements taken across 20 heart beats at one specific location in the MPA, which were averaged over the heart beats to obtain a single average time series covering one heart beat (the averaged blood flow and data can be seen in Figure 1 (C)). The raw data for the 20 heart beats are not available. The flow measurements are used as inflow in the MPA (boundary condition for the PDEs). The parameter estimation is based on the pressure measurements.

2.4 | Synthetic data

We simulate one data set from the mathematical model with parameter values consistent with the physiological data (i.e., the posterior median values), and to these data we add iid additive Gaussian noise, with the noise variance being the posterior median found based on the real data.

3 | REVIEW OF RELEVANT STATISTICAL METHODOLOGY

This section provides a brief review of relevant methodological background. Our methodological innovation is presented in the next section, Section 4.

3.1 | Gaussian processes

We use a regression GP model for statistical emulation of the unnormalised log posterior distribution[11,72,73] (see Section 4), and a classification GP model to decide on the success of a PDE evaluation for particular parameter configurations. A GP model is a non-parametric Bayesian technique, which we briefly review in Appendix A.2. More details on GPs can be found in Reference 74.

3.2 | Gradient-based MCMC algorithms

This section summarises the gradient-based MCMC algorithms used in this study. We also refer to Table 2, which provides an overview of these algorithms and outlines the differences between them.

HMC . HMC[22] is a powerful MCMC scheme which suppresses the random walk behaviour of Metropolis-Hastings MCMC by introducing an auxiliary variable, the "momentum" variable, $\mathbf{r} \in \mathbb{R}^{d \times 1}$, with density $p(\mathbf{r}) = \mathcal{N}(\mathbf{r} \,|\, \mathbf{0}, \mathbf{M})$, which guides the search towards high posterior density regions. HMC simulates Hamiltonian dynamics by using gradient information from the log target density.

Defining H: Hamiltonian, $\mathbf{r}$: "momentum" vector, and $\boldsymbol{\theta}$: "position" vector (representing our parameters that.
we want to infer), the Hamiltonian dynamic equations are:

$$\frac{d\theta_i}{dt} = \frac{\partial H}{\partial r_i}, \quad \frac{dr_i}{dt} = -\frac{\partial H}{\partial \theta_i}. \tag{6}$$

The Hamiltonian equations leave the Hamiltonian invariant, that is, $H(\boldsymbol{\theta}^*, \mathbf{r}^*) = H(\boldsymbol{\theta}, \mathbf{r})$,

TABLE 2 Overview of the Hamiltonian and Lagrangian Monte Carlo algorithms used: Adaptive Hamiltonian Monte Carlo (AHMC), No U-turn sampler (NUTS), Adaptive Riemann Manifold HMC (ARMHMC), Adaptive Lagrangian Dynamical Monte Carlo (ALDMC)

	Auxiliary			2nd and 3rd	
Algorithm	variables	Integration scheme	Parameter tuning method	order method	"mass matrix"
AHMC	"momentum"	Explicit leapfrog scheme	BO	No	Identity matrix
NUTS	"momentum"	Explicit leapfrog scheme	Stochastic optimisation for ϵ, L chosen recursively	No	Identity matrix
ARMHMC	"momentum"	Implicit generalised leapfrog scheme	BO	Yes	Negative Hessian matrix of log posterior
ALDMC	"velocity"	Explicit generalised leapfrog scheme	BO	Yes	Negative Hessian matrix of log posterior

Note: These algorithms augment the parameter space with an auxiliary variable. AHMC, NUTS and ARMHMC use "momentum" as an auxiliary variable, while ALDMC uses "velocity." These algorithms simulate Hamiltonian or Lagrangian dynamics, that are numerically integrated. The integration scheme is based on the leapfrog method with a small step size ϵ and a number of steps L (called tuning parameters). The adaptive algorithms use Bayesian optimisation (BO) for parameter tuning, while NUTS has a different mechanism, see Section 3.2 for details. AHMC and NUTS use the first order derivative of the log posterior, while ARMHMC and ALDMC use also the second and third order derivatives. Thus, AHMC and NUTS set the "mass matrix" for the auxiliary variable (its covariance matrix) to the identity matrix, while for ARMHMC and ALDMC, the mass matrix is set to the negative Hessian matrix of the log posterior. The terms "momentum," "velocity" and "mass matrix" are used for their mathematical equivalence to the corresponding terms in Hamiltonian mechanics.

$$\frac{dH}{dt} = \sum_{i=1}^{d}\left(\frac{\partial H}{\partial r_i}\frac{dr_i}{dt} + \frac{\partial H}{\partial \theta_i}\frac{d\theta_i}{dt}\right) = \sum_{i=1}^{d}\left(\frac{\partial H}{\partial r_i}\left(-\frac{\partial H}{\partial \theta_i}\right) + \frac{\partial H}{\partial \theta_i}\frac{\partial H}{\partial r_i}\right) = 0. \tag{7}$$

The Hamiltonian dynamics (Equation (6)) are numerically integrated for a specified fictitious time. The leapfrog integrator[22] is used, in which time is discretised using a small step size, $\epsilon > 0$, and the trajectory is run for a number of leapfrog steps, L. The numerical integration induces an error, which implies that $H(\boldsymbol{\theta}^*, \mathbf{r}^*)$ no longer equals $H(\boldsymbol{\theta}, \mathbf{r})$.

Further defining $\mathbf{M}$: "mass matrix" for the "momentum" (its covariance matrix), E: "potential energy," K: "kinetic energy," we have:

$$E(\boldsymbol{\theta}) = -(\log p(\boldsymbol{\theta}) + \log p(\mathbf{y}|\boldsymbol{\theta})); \quad K(\mathbf{r}) = \frac{\mathbf{r}^T\mathbf{M}^{-1}\mathbf{r}}{2}; \quad H(\boldsymbol{\theta},\mathbf{r}) = E(\boldsymbol{\theta}) + K(\mathbf{r}), \tag{8}$$

where $\log p(\boldsymbol{\theta})$ and $\log p(\mathbf{y}|\ \boldsymbol{\theta})$ are the log prior distribution of the parameters and the log data likelihood.

Note that the terms "mass matrix," "momentum," as well as "kinetic" and "potential energy" are used for their mathematical equivalence to the corresponding terms in Hamiltonian mechanics.

In statistical terms, the joint distribution is

$$p(\boldsymbol{\theta},\mathbf{r}|\mathbf{y}) = p(\boldsymbol{\theta}\,|\,\mathbf{r},\mathbf{y})p(\mathbf{r}) = p(\boldsymbol{\theta}|\mathbf{y})\mathcal{N}(\mathbf{r}\,|\,\mathbf{0},\mathbf{M}) \propto \exp(-E(\boldsymbol{\theta}))\exp(-K(\mathbf{r})) \propto \exp(-H(\boldsymbol{\theta},\mathbf{r})). \tag{9}$$

Thus, if we denote the log posterior of the parameters we want to infer $\boldsymbol{\theta}$ as $\log p(\boldsymbol{\theta}|\ \mathbf{y})$, then the negative log auxiliary joint distribution is:

$$H(\boldsymbol{\theta},\mathbf{r}|\mathbf{y}) = -\log p(\boldsymbol{\theta}|\mathbf{y}) + \frac{1}{2}\log\left((2\pi)^d|\mathbf{M}|\right) + \frac{\mathbf{r}^T\mathbf{M}^{-1}\mathbf{r}}{2} + \log Z, \tag{10}$$

where Z is a normalising constant and $|\mathbf{M}|$ is the determinant of the "mass matrix" $\mathbf{M}$.

In HMC, at the end of each leapfrog trajectory (defined as a segment between 2 subsequent acceptance steps), a new point is proposed, and accepted with probability min(1, $\exp(-H(\boldsymbol{\theta}^*, \mathbf{r}^*|\ \mathbf{y}) + H(\boldsymbol{\theta}, \mathbf{r}|\ \mathbf{y}))$. If the error from the numerical integration of the Hamiltonian dynamics in Equation (6) is small, then the acceptance probability will be high. If the new point is rejected, we keep the current point. The next trajectory is then simulated, and each trajectory starts with the resampling of the "momentum" variables from their marginal distribution $\mathcal{N}(\mathbf{r} \mid \mathbf{0}, \mathbf{M})$, to allow properly being integrated out from the joint distribution, that is,

$$p(\boldsymbol{\theta}|\mathbf{y}) = \int p(\boldsymbol{\theta} \mid \mathbf{r}, \mathbf{y}) p(\mathbf{r}) d\mathbf{r}. \tag{11}$$

The HMC algorithm produces an ergodic, time reversible Markov chain, which satisfies detailed balance and whose stationary distribution is the marginal distribution $p(\boldsymbol{\theta}| \mathbf{y})$.

In the classical HMC algorithm, $\mathbf{M}$ is kept fixed at the identity matrix, and the HMC tuning parameters, ϵ and L, are hand-tuned in to get an acceptance rate >65%[22] and a large effective sample size.

Below we summarise extensions to the HMC algorithm, aimed at improving the algorithm's performance by automatic tuning of the HMC parameters or of the mass matrix $\mathbf{M}$, to which the algorithm is known to be highly sensitive.

RMHMC. Riemann Manifold HMC (RMHMC)[63] is an improved version of HMC, as it exploits the Riemannian geometry of the parameter space. It makes use of the curvature of the target distribution to set $\mathbf{M}$, that is, $\mathbf{M}$ changes within every trajectory to adapt to the target density curvature ($\mathbf{M} \rightarrow \mathbf{M}(\boldsymbol{\theta})$ in Equation (10)). We set $\mathbf{M}$ to be the observed Fisher Information matrix (the matrix of negative second order derivatives of the log likelihood), plus the negative Hessian matrix of the log prior. The joint distribution in Equation (9) is no longer factorisable. An implicit integrator is used (the Generalised Leapfrog algorithm), as proposals generated from the Leapfrog integrator no longer satisfy detailed balance in HMC: $\mathbf{M}(\boldsymbol{\theta}(t)) \neq \mathbf{M}(\boldsymbol{\theta}(t + \epsilon))$. The implicit integrator incurs high numerical costs. In RMHMC, ϵ and L are fixed.

LDMC. To overcome the increased computational costs in RMHMC associated with iteratively solving the equations of the implicit integrator, Lagrangian dynamics can be used instead of Hamiltonian dynamics. This leads to the Lagrangian Dynamical Monte Carlo algorithm - LDMC.[64] LDMC uses an explicit geometric integrator that replaces the "momentum" variable in RMHMC by "velocity" (used in analogy with classical mechanics), which improves the computational efficiency. However, the volume in phase space is no longer preserved, hence the Jacobian transformation is needed to adjust the acceptance probability to ensure detailed balance.[64] In LDMC, $\mathbf{M}$ is adjusted to the curvature of the posterior distribution at every step throughout the trajectory, and ϵ and L are kept fixed.

NUTS. The No U-turn sampler (NUTS), proposed in Reference 62, chooses L recursively by moving in parameter space until the HMC trajectory starts to double back and retrace its steps. The points collected along the way are then sampled in a way that ensures detailed balance. The algorithm adapts ϵ in the burn-in phase by means of a stochastic optimisation algorithm (the primal-dual averaging). $\mathbf{M}$ is kept fixed at the identity matrix.

AHMC. Adaptive HMC (AHMC), proposed in Reference 65, uses Bayesian Optimisation[75] to tune ϵ and L, while allowing for infinite adaptation of the parameters, with the adaptation becoming less likely as the number of iterations increases. $\mathbf{M}$ is kept fixed at the identity matrix. In AHMC, an objective function dependent on ϵ and L is chosen, for example, the expected squared jumping distance (ESJD) normalised by the number of leapfrog steps[65]:

$$\frac{\mathbb{E}^{\epsilon, L}_{p(.)} \left\| \boldsymbol{\theta}^{(t+1)} - \boldsymbol{\theta}^{(t)} \right\|^2}{\sqrt{L}}. \tag{12}$$

The normalised ESJD is an expensive black-box function, due to the intractable expectation with respect to the target distribution $p(.)$, so it is approximated by an empirical estimator. The idea of emulation and Bayesian optimisation[76] is adopted; the normalised ESJD is maximised by constructing a surrogate objective function using Gaussian Processes, called acquisition function. The problem is turned into the maximisation of this computationally cheap acquisition function. The same procedure can be used for the automatic tuning of the step size and the number of leapfrog steps in RMHMC or LDMC.

4 | NOVEL STATISTICAL METHODOLOGY: COMBINATION OF MCMC WITH GAUSSIAN PROCESSES IN A DELAYED ACCEPTANCE FRAMEWORK

The application of Bayesian inference to complex physiological models poses several challenges, and we mention a few:

1. We need an efficient and accurate method for inferring the unknown parameters and quantifying their uncertainty from limited and noisy data.
2. It is expensive to obtain the data likelihood as it requires numerically integrating a system of PDEs.
3. We have a priori unknown constraints on the joint parameter space, that is, regions in the joint parameter space that violate the assumptions of the mathematical model.

First challenge. To address the first issue, we implement a series of state-of-the-art MCMC methods (described in Section 3.2 and summarised in Table 2) and pick the method that is best in terms of accuracy and computational efficiency. Our methods include: Hamiltonian Monte Carlo, No U-turn sampler, Riemann Manifold HMC and Lagrangian Dynamical Monte Carlo. These algorithms are combined with Bayesian optimisation for automatic performance tuning. The comparison between all of these algorithms is novel in the context of biophysical modelling.

Second challenge. Rasmussen addresses the second issue by combining MCMC (the HMC algorithm) with emulation using GPs as part of the GPHMC algorithm.[32] Our contribution is to further extend and improve GPHMC.

We start by emphasising the need for GPHMC. Algorithm 1a compares the HMC algorithm (textbook knowledge) to Rasmussen's GPHMC algorithm in terms of the number of PDE evaluations required to calculate the data log likelihood and its gradient, which is included in the caption of the algorithm. Specific differences between HMC and GPHMC are marked by the text in blue colour. Algorithm 1a is just a conceptual outline; the reader is referred to Algorithms 1b and 1c for a detailed pseudocode of our extended GPHMC algorithm. Algorithm 1a in words: HMC on the true log likelihood incurs excessive computational costs due to the need to numerically solve the PDEs of the biophysical model at every leapfrog integration step of the Hamiltonian dynamics (Equation (6)) for a large number of HMC samples. The computational costs can be reduced substantially by moving the HMC scheme to the surrogate space defined by the statistical emulator. In that case, we avoid evaluating the PDEs for each of the L steps along the trajectory, instead we only evaluate the PDEs once per HMC sample (at the end point of the Hamiltonian trajectory) to obtain the true log likelihood, used in the decision step; the surrogate log likelihood (as predicted by the emulator) is used to guide the search along the trajectory. An HMC trajectory typically has in the order of $L = 100\text{–}1000$ steps, which if carried out in the original space would require in the order of $100 \times (d + 1)$ to $1000 \times (d + 1)$ (d: number of parameters) PDE evaluations per HMC sample, thus a reduction in the computational complexity by about two to three orders of magnitude is obtained. The term $d + 1$ is the sum of one PDE evaluation to find the log likelihood, and d PDE evaluations to find the numerical derivatives by a first-order differencing scheme with respect to each of the d parameters. When HMC is run in the original space, several aspects regarding the calculation of the numerical derivatives of the simulated log posterior must be considered, see Appendix A.3 for details. In contrast, GPHMC requires derivatives of the GP predictive mean and variance (instead of the numerical derivatives for the simulator), which can be obtained

Algorithm 1

Conceptual outline for Hamiltonian Monte Carlo (HMC) versus HMC coupled with emulation using Gaussian Processes (GPHMC) algorithm. The total number of model (PDE) evaluations required for running each algorithm is: HMC – SL(d + 1) versus GPHMC – S

```
Define a $d$-dimensional vector $\boldsymbol{\theta}$ with $\theta_k$ the $k^{th}$ element, $k = 1...d$; S: number of HMC samples; L: number of HMC trajectory steps; $p(\mathbf{y}|\ \boldsymbol{\theta})$: simulator data likelihood (Equation (18))
for i = 1: S do.
    for j = 1: L do.
        Solve PDEs to get $\log p(\mathbf{y}|\ \boldsymbol{\theta}^j)$ and $\frac{\partial \log p(\mathbf{y}|\boldsymbol{\theta}^j)}{\partial \theta^j_k}$
    end for.
    Propose $\boldsymbol{\theta}^L$.
    Solve PDEs to get $\log p(\mathbf{y}|\ \boldsymbol{\theta}^L)$ and accept/reject in a M-H step.
end for.
```

```
Define a $d$-dimensional vector $\boldsymbol{\theta}$ with $\theta_k$ the $k^{th}$ element, $k = 1d$; S: number of HMC samples; L: number of HMC trajectory steps; $\tilde{p}(\mathbf{y}|\boldsymbol{\theta})$: emulator data likelihood (Equation (19)).
for i = 1: S do.
    for j = 1: L do.
        Use GPs to predict $\log\tilde{p}(\mathbf{y}|\boldsymbol{\theta}^j)$ and $\frac{\partial \log\tilde{p}(\mathbf{y}|\boldsymbol{\theta}^j)}{\partial \theta^j_k}$.
    end for
    Propose $\boldsymbol{\theta}^L$
    Solve PDEs to get $\log p(\mathbf{y}|\ \boldsymbol{\theta}^L)$ and accept/reject in a M-H step.
end for.
```

Algorithm 2

GP HMC algorithm with DA – initial and exploratory phase

```
1: Let $D=\{(\mathbf{X}_e,\mathcal{S}_e):\lambda=1\}$: training set for the GP emulator and $\mathcal{H}=(\mathbf{X},\boldsymbol{\lambda})$: training set for the GP classifier, where $\mathbf{X}$: $n\times d$ matrix of input parameter vectors $\boldsymbol{\theta}$, and $\mathbf{X}_e$: subset of $\mathbf{X}$, $\mathcal{S}_e$: vector of residual sum-of-square (RSS) values, and $\boldsymbol{\lambda}$: $n\times 1$ vector of success labels (0,1). Denote $S$: number of HMC samples, L: number of HMC trajectory steps, $\epsilon$: step size, $\mathcal{S}(\boldsymbol{\theta})$: simulator RSS, $\tilde{\mathcal{S}}(\boldsymbol{\theta})$: emulator RSS, $\mathbf{M}$: mass matrix, $f(.)$: emulated RSS function, and $p(\mathbf{y}|\boldsymbol{\theta})$: simulator data likelihood, $\tilde{p}(\mathbf{y}|\boldsymbol{\theta})$: emulator data likelihood, $E(\boldsymbol{\theta})$: true potential function, $\tilde{E}(\boldsymbol{\theta})$: surrogate potential function, $K(\mathbf{r})$: kinetic energy for the momentum variable $\mathbf{r}$.
2: $\log p(\mathbf{y}|\boldsymbol{\theta})=-\frac{\mathcal{S}(\boldsymbol{\theta})}{2\sigma^2}-\frac{n}{2}\log(2\pi\sigma^2)$ and $\log\tilde{p}(\mathbf{y}|\boldsymbol{\theta})=-\frac{\tilde{\mathcal{S}}(\boldsymbol{\theta})}{2\sigma^2}-\frac{n}{2}\log(2\pi\sigma^2)$ for $\epsilon \underset{\sim}{iid} \mathcal{MVN}(\mathbf{0},\sigma^2\mathbf{I})$; $E(\boldsymbol{\theta})=-(\log p(\mathbf{y}|\boldsymbol{\theta})+\log p(\boldsymbol{\theta}))$, where $p(\boldsymbol{\theta})$: prior distribution; $\tilde{E}(\boldsymbol{\theta})=-(\log\tilde{p}(\mathbf{y}|\boldsymbol{\theta})+\log\tilde{p}(\boldsymbol{\theta}))$, where $\tilde{p}(\boldsymbol{\theta})$: modified prior distribution (see Equation (17)).
3: **INITIAL DESIGN STAGE**: Build the GP emulator and the GP classifier.
4: **EXPLORATORY PHASE**: Set $\boldsymbol{\theta}^0$ and $l=1$, where $l$ marks the $l^{th}$ point being deleted.
5: **for** i = 1: S **do** ▷ loop over HMC samples
6:     Draw $\mathbf{r}\sim\exp(-K(\mathbf{r}))$ and let $\boldsymbol{\theta}_0=\boldsymbol{\theta}^{i-1}$ and $\mathbf{r}_0=\mathbf{r}+\frac{\epsilon}{2}\frac{\partial\tilde{E}}{\partial\boldsymbol{\theta}}\Big|_{\boldsymbol{\theta}_0}$
7:     **for** j = 1: L **do** ▷ loop over HMC steps
8:         $\boldsymbol{\theta}_j=\boldsymbol{\theta}_{j-1}+\epsilon\mathbf{M}^{-1}\mathbf{r}_{j-1}$; $\mathbf{r}_j=\mathbf{r}_{j-1}+\epsilon\frac{\partial\tilde{E}}{\partial\boldsymbol{\theta}}\Big|_{\boldsymbol{\theta}_j}$
9:     **end for**
10:    $\mathbf{r}_L=\mathbf{r}_{L-1}+\frac{\epsilon}{2}\frac{\partial\tilde{E}}{\partial\boldsymbol{\theta}}\Big|_{\boldsymbol{\theta}_L}$
11:    Set proposed points $(\boldsymbol{\theta}^*,\mathbf{r}^*)=(\boldsymbol{\theta}_L,\mathbf{r}_L)$ and $\tilde{\mathcal{S}}(\boldsymbol{\theta}^*)=\mathbb{E}(f(\boldsymbol{\theta}^*)|D)-\sqrt{\mathrm{var}(f(\boldsymbol{\theta}^*)|D)}$, where $\mathbb{E}(f(\boldsymbol{\theta}^*)|D)$ and $\sqrt{\mathrm{var}(f(\boldsymbol{\theta}^*)|D)}$ are the GP posterior predictive mean and standard deviation (Equations (A4) and (A5))
12:    M-H accept/reject step with 1st stage acceptance probability (emulator based): $\alpha_1(\boldsymbol{\theta}^*,\mathbf{r}^*|\boldsymbol{\theta}^{i-1},\mathbf{r})=1\wedge\frac{\exp(-\tilde{E}(\boldsymbol{\theta}^*))}{\exp(-\tilde{E}(\boldsymbol{\theta}^{i-1}))}\frac{\exp(-K(\mathbf{r}^*))}{\exp(-K(\mathbf{r}))}$, with $\tilde{E}$ computed from $\tilde{\mathcal{S}}$ as explained in line 2.
13:    **if** $\alpha_1\geq v_1,\quad v_1\sim U(0,1)$ **then**, solve PDEs for $\boldsymbol{\theta}^*=\boldsymbol{\theta}_L$ to get $(\boldsymbol{\theta}^*,\mathcal{S}(\boldsymbol{\theta}^*),\lambda^*)$ - see Sections 4 and 5.
14:    Calculate the 2nd stage acceptance probability (simulator based): $\alpha_2(\boldsymbol{\theta}^*|\boldsymbol{\theta}^{i-1})=1\wedge\frac{\exp(-E(\boldsymbol{\theta}^*))}{\exp(-E(\boldsymbol{\theta}^{i-1}))}\frac{\exp(-\tilde{E}(\boldsymbol{\theta}^{i-1}))}{\exp(-\tilde{E}(\boldsymbol{\theta}^*))}$, with $E$ computed from $\mathcal{S}$ as explained in line 2
15:        **if** $\alpha_2\geq v_2,\quad v_2\sim U(0,1)$ **then** set $\boldsymbol{\theta}^i=\boldsymbol{\theta}^*$.
16:            **if** $\mathcal{S}(\boldsymbol{\theta}^l)>T$, where $T$: threshold value chosen based on the $\mathcal{S}$ values from the initial design stage (e.g. $T$ = 10th percentile) **then** update $D=D\backslash(\boldsymbol{\theta}^l,\mathcal{S}(\boldsymbol{\theta}^l))$, $\mathcal{H}=\mathcal{H}\backslash(\boldsymbol{\theta}^l,\lambda^l)$.
17:            **else** set l = l + 1.
18:            **end if**.
19:            Re-train GP emulator with $D=D\cup(\boldsymbol{\theta}^*,\mathcal{S}(\boldsymbol{\theta}^*))$ and GP classifier with $\mathcal{H}=\mathcal{H}\cup(\boldsymbol{\theta}^*,\lambda^*)$
20:        **else**
21:            $\boldsymbol{\theta}^i=\boldsymbol{\theta}^{i-1}$
22:        **end if**.
23:    **else**
24:        $\boldsymbol{\theta}^i=\boldsymbol{\theta}^{i-1}$
25:    **end if**.
26: **end for**.
27: Update $D=D\backslash(\boldsymbol{\theta},\mathcal{S}(\boldsymbol{\theta}))$, $\mathcal{H}=\mathcal{H}\backslash(\boldsymbol{\theta},\lambda)$ for the remaining $\boldsymbol{\theta}$ for which $\mathcal{S}(\boldsymbol{\theta})>T$.
28: Re-train GP emulator with new $D$ and GP classifier with new $\mathcal{H}$ and enter the **SAMPLING PHASE**.
```

analytically (Equations (A6) and (A7)). Additionally, for GPHMC, the smoothness, thus differentiability of the log posterior is controlled via the GP kernel (e.g., the squared exponential kernel, which is used in this study, is infinitely differentiable), while for conventional HMC, the log posterior might not be differentiable everywhere, see Appendix A.3.

We use the emulator for the proposal move. The M-H acceptance/rejection decision at the end of the trajectory is based on the true posterior distribution, the surrogate posterior distribution and the proposal probability ratio from the

Algorithm 3

GP HMC algorithm with DA – sampling phase

```
1: Denote $\boldsymbol{\theta}$: input parameter vector, $S$: number of HMC samples, L: number of HMC trajectory steps, $\epsilon$: step size, $\mathcal{S}(\boldsymbol{\theta})$: simulator residual sum-of-square (RSS) value, $\tilde{\mathcal{S}}(\boldsymbol{\theta})$: emulator RSS, $\mathbf{M}$: mass matrix, $f(.)$: emulated RSS function, $p(\mathbf{y}|\boldsymbol{\theta})$: simulator data likelihood, $\tilde{p}(\mathbf{y}|\boldsymbol{\theta})$: emulator data likelihood, $E(\boldsymbol{\theta})$: true potential function, $\tilde{E}(\boldsymbol{\theta})$: surrogate potential function, $K(\mathbf{r})$: kinetic energy for the momentum variable $\mathbf{r}$.
2: $\log p(\mathbf{y}|\boldsymbol{\theta}) = -\frac{\mathcal{S}(\boldsymbol{\theta})}{2\sigma^2} - \frac{n}{2}\log(2\pi\sigma^2)$ and $\log\tilde{p}(\mathbf{y}|\boldsymbol{\theta}) = -\frac{\tilde{\mathcal{S}}(\boldsymbol{\theta})}{2\sigma^2} - \frac{n}{2}\log(2\pi\sigma^2)$ for $\epsilon iid\mathcal{MVN}(\mathbf{0},\sigma^2\mathbf{I})$; $E(\boldsymbol{\theta}) = -(\log p(\mathbf{y}|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta}))$, where $p(\boldsymbol{\theta})$: prior distribution; $\tilde{E}(\boldsymbol{\theta}) = -(\log\tilde{p}(\mathbf{y}|\boldsymbol{\theta}) + \log\tilde{p}(\boldsymbol{\theta}))$, where $\tilde{p}(\boldsymbol{\theta})$: modified prior distribution (see Equation (17)).
3: Initialise $\boldsymbol{\theta}^0$.
4: for i = 1: S do                                              ▷ loop over HMC samples
5:    Draw $\mathbf{r} \sim \exp(-K(\mathbf{r}))$.
6:    Let $\boldsymbol{\theta}_0 = \boldsymbol{\theta}^{i-1}$ and $\mathbf{r}_0 = \mathbf{r} + \frac{\epsilon}{2}\frac{\partial\tilde{E}}{\partial\boldsymbol{\theta}}\Big|_{\boldsymbol{\theta}_0}$.
7:    for j = 1: L do                                           ▷loop over HMC steps
8:       $\boldsymbol{\theta}_j = \boldsymbol{\theta}_{j-1} + \epsilon\mathbf{M}^{-1}\mathbf{r}_{j-1}$
9:       $\mathbf{r}_j = \mathbf{r}_{j-1} + \epsilon\frac{\partial\tilde{E}}{\partial\boldsymbol{\theta}}\Big|_{\boldsymbol{\theta}_j}$
10:   end for
11:   $\mathbf{r}_L = \mathbf{r}_{L-1} + \frac{\epsilon}{2}\frac{\partial\tilde{E}}{\partial\boldsymbol{\theta}}\Big|_{\boldsymbol{\theta}_L}$
12:   Set proposed points $(\boldsymbol{\theta}^*, \mathbf{r}^*) = (\boldsymbol{\theta}_L, \mathbf{r}_L)$ and $\tilde{S}(\theta^*) = \mathrm{E}(f(\theta^*)|D)$, where $\mathrm{E}(f(\theta^*)|D)$is the GP posterior predictive mean (see Equation (A4))
13:   Compute $\tilde{E}$ from $\tilde{\mathcal{S}}$ as explained in line 2.
14:   M-H accept/reject step with 1st stage acceptance probability (emulator based):

      $\alpha_1(\boldsymbol{\theta}^*,\mathbf{r}^*\,|\,\boldsymbol{\theta}^{i-1},\mathbf{r}) \;=\; 1\wedge\frac{\exp(-\tilde{E}(\boldsymbol{\theta}^*))}{\exp(-\tilde{E}(\boldsymbol{\theta}^{i-1}))}\frac{\exp(-K(\mathbf{r}^*))}{\exp(-K(\mathbf{r}))}$

15:   if $\alpha_1 \geq v_1$,   $v_1 \sim U(0, 1)$ then.
16:      Solve PDEs for $\boldsymbol{\theta}^*$ to get $(\boldsymbol{\theta}^*,\mathcal{S}(\boldsymbol{\theta}^*),\lambda^*)$ – see Sections 4 and 5.
17:      Compute $E$ from $\mathcal{S}$ as explained in line 2.
18:      Calculate the 2nd stage acceptance probability (simulator based):

      $\alpha_2(\boldsymbol{\theta}^*|\boldsymbol{\theta}^{i-1}) \;=\; 1\wedge\frac{\exp(-E(\boldsymbol{\theta}^*))}{\exp(-E(\boldsymbol{\theta}^{i-1}))}\frac{\exp(-\tilde{E}(\boldsymbol{\theta}^{i-1}))}{\exp(-\tilde{E}(\boldsymbol{\theta}^*))}$

19:      if $\alpha_2 \geq v_2$,   $v_2 \sim U(0, 1)$ then.
20:         Set $\boldsymbol{\theta}^i = \boldsymbol{\theta}^*$
21:      else
22:         $\boldsymbol{\theta}^i = \boldsymbol{\theta}^{i-1}$
23:      end if.
24:   else
25:      $\boldsymbol{\theta}^i = \boldsymbol{\theta}^{i-1}$
26:   end if.
27: end for.
```

HMC dynamics on the emulated posterior distribution space, as shown in Equations (15) and (16). If the emulator is a poor representation of the simulator, the acceptance rate at the end of the HMC trajectory will go down, as the high probability areas in the true and the surrogate space do not match. However, the algorithm is still mathematically guaranteed to converge to the true posterior distribution, albeit at a lower convergence rate.[32] Hence, a suboptimal emulator will not affect the mathematical accuracy of the scheme, but merely its computational efficiency.

To avoid unnecessarily solving PDEs and to potentially speed up simulations, we extend the GPHMC algorithm by introducing Delayed Acceptance[53,55] HMC, with the following idea:

DA. The Delayed Acceptance (DA) algorithm, proposed in Reference 53, and slightly modified by Sherlock et al.,[54,55] uses a two-stage acceptance procedure, with two separate acceptance/rejection decisions. The idea is that the first decision is a computationally fast pre-filter step, which upon rejection of a proposed new parameter avoids having to carry out the computationally expensive second step. As in References 54 and 55, the first acceptance probability based on the emulator is given in Equation (13) and the second acceptance probability is expressed in Equation (14):

$$\alpha_1(\boldsymbol{\theta}^*|\boldsymbol{\theta}) = 1 \wedge \frac{\tilde{p}(\boldsymbol{\theta}^*|\mathbf{y})q(\boldsymbol{\theta}|\boldsymbol{\theta}^*)}{\tilde{p}(\boldsymbol{\theta}|\mathbf{y})q(\boldsymbol{\theta}^*|\boldsymbol{\theta})}, \tag{13}$$

$$\alpha_2(\boldsymbol{\theta}^*|\boldsymbol{\theta}) = 1 \wedge \frac{p(\boldsymbol{\theta}^*|\mathbf{y})}{p(\boldsymbol{\theta}|\mathbf{y})}\frac{\tilde{p}(\boldsymbol{\theta}|\mathbf{y})}{\tilde{p}(\boldsymbol{\theta}^*|\mathbf{y})}, \tag{14}$$

where $\tilde{p}(.)$ is the approximate posterior distribution, constructed using the emulator, $p(.)$ is the exact posterior distribution, obtained using the simulator, and $q(.)$ is the proposal distribution. We have used the shorthand notation $a \wedge b = \min\{a, b\}$. The algorithm preserves detailed balance with respect to the posterior distribution $p(\boldsymbol{\theta}|\,\mathbf{y})$.[53,55]

For HMC-type algorithms, Equations (13) and (14) translate to:

$$\alpha_1^{\mathrm{HMC}}(\boldsymbol{\theta}^*,\mathbf{r}^*|\boldsymbol{\theta},\mathbf{r}) = 1 \wedge \frac{\exp\left(-\tilde{H}[\boldsymbol{\theta}^*,\mathbf{r}^*]\right)1}{\exp\left(-\tilde{H}[\boldsymbol{\theta},\mathbf{r}]\right)1} = 1 \wedge \frac{\exp\left(-\tilde{H}\boldsymbol{\theta}^*,\mathbf{r}^*]\right)}{\exp\left(-\tilde{H}[\boldsymbol{\theta},\mathbf{r}]\right)}, \tag{15}$$

where $\tilde{H}(\boldsymbol{\theta},\mathbf{r}) = \tilde{E}(\boldsymbol{\theta}) + K(\mathbf{r})$ is the Hamiltonian function. Also, $q(.|\,.) = 1$ is the proposal distribution, which comes from following a set of Hamiltonian dynamics steps, that are deterministic (i.e., the proposal is a Dirac delta function at the proposed point).

$$\alpha_2^{\mathrm{HMC}}(\boldsymbol{\theta}^*,\mathbf{r}^*|\boldsymbol{\theta},\mathbf{r}) = 1 \wedge \frac{\exp(-E(\boldsymbol{\theta}^*))}{\exp(-E(\boldsymbol{\theta}))}\frac{\exp\left(-\tilde{E}(\boldsymbol{\theta})\right)}{\exp\left(-\tilde{E}(\boldsymbol{\theta}^*)\right)}. \tag{16}$$

The computational gain is related to the fact that we only need to carry out the computationally expensive numerical integration of the PDEs if the proposed parameter vector is accepted according to the emulator.

Third challenge. The third challenge in our study is addressed by constructing a GP classifier that automatically learns the infeasible parameter regions. To apply the GPHMC scheme to parameter estimation in a complex biophysical model, we typically need to deal with the fact that for certain parameter values and combinations, the underlying physical assumptions of the model are violated, or the solver used is inappropriate for the problem, producing no outputs from the simulation software. While the second matter can be tackled by for example, trying a different solver, decreasing the convergence threshold value, or increasing the number of discretisation steps, there is no clear resolution for the first issue. This may be caused by the inappropriateness of the parameter ranges, which are generally chosen by varying one parameter at a time, while fixing all the other parameters to biologically realistic values. However, when parameters are simultaneously changed, given for example, the fixed network geometry and inflow, the resulting parameter combinations may break physiological assumptions (e.g., a large arterial stiffness may not be compatible with high compliance downstream). We stress that a multivariate classifier should only be implemented upon thoroughly checking the suitability of the solver. For parameter values in the "invalid" domain, different solvers may be tried. If the crash is independent of the solver used, this suggests that the crash is of a more fundamental nature (i.e., violation of the physical model assumptions).

In a standard MCMC simulation based on the actual model, an invalid parameter vector can be assigned zero likelihood. Hence, if such a parameter vector is proposed, it will be rejected in the Metropolis-Hastings acceptance/rejection step. However, dealing with such invalid regions in the context of emulation requires some extra care. A naive and straightforward approach is to set the likelihood for an invalid parameter vector to a very small value close to zero (i.e., the log likelihood to a negative value with large modulus) when training the GP emulator. However, this approach is unlikely to lead to a good emulator. A sudden shift to an extreme value will drive the lengthscale of the GP kernel to a very small value in the hyperparameter estimation step, which in turn will cause ripples in the GP interpolant and hence overfitting in the valid regime. We address this issue by introducing a GP classifier. Let λ denote a binary variable to indicate if the parameter vector $\boldsymbol{\theta}$ falls into a valid regime ($\lambda = 1$) or invalid regime ($\lambda = 0$). Given a set of parameter vector - label pairs obtained during the initial design and exploration phase (see below),

$$\mathscr{H} = \{(\boldsymbol{\theta}_1,\lambda_1),\dots,(\boldsymbol{\theta}_n,\lambda_n)\}$$

we can train a GP classifier, as reviewed in Appendix A.2.2, to predict the probability $p(\lambda = 1 | \boldsymbol{\theta}, \mathscr{H})$. We can now use these probabilities to modify the prior:

$$\tilde{p}(\boldsymbol{\theta}) = p(\boldsymbol{\theta})p(\lambda=1|\boldsymbol{\theta},\mathscr{H})/Z, \tag{17}$$

where $Z = \int p(\boldsymbol{\theta})p(\lambda = 1 | \boldsymbol{\theta}, \mathscr{H})d\boldsymbol{\theta}$ is a normalisation constant, which cancels out in the Metropolis-Hastings acceptance/rejection step. When running in the emulated space, the sampler uses the modified prior $\tilde{p}(\boldsymbol{\theta})$ instead of the original prior $p(\boldsymbol{\theta})$ to avoid moving into invalid parameter regions (where the probability of success $p(\lambda = 1 | \boldsymbol{\theta}, \mathscr{H})$ is low).

In summary, in the exploration phase, the GP classifier is trained on the valid and invalid parameter vectors, that is, parameter vectors with (label 1) and without (label 0) software outputs. The predicted probability of failure/success is then included in the prior probability on the parameter space. In this way, moves into the invalid parameter regime are discouraged by the GP classifier, and the GP emulator does not need to provide accurate predictions here.¶ We note that a similar idea of combining a GP classifier with a GP emulator has been proposed in the context of Bayesian optimisation with unknown constraints.[77,78]

GPHMC algorithm coupled with a GP classifier and Delayed Acceptance.

We now put all the elements explained above together. Thus, we provide a description of the implementation of the DA-GPHMC algorithm, with our adaptation to mathematical models with limited valid domains of a priori unknown location. Pseudocode for GPHMC coupled with a GP classifier and DA can be found in Algorithms 1b and 1c. Specific equations used in the practical implementation of the algorithm are given in the Simulations Section 5. A diagram summarising Rasmussen's GPHMC algorithm can be seen in Figure 2, and in Figure 3 we illustrate the workflow of our proposed DA-GPHMC method, where the latter algorithm is used within the former.

- *Initial design stage*. Starting from a space filling design in parameter space to capture the compact support of the biophysical parameters (see Section 2.1 for more details), for example, using a Latin hypercube[79] or a Sobol sequence,[80] integrate the PDEs numerically for each parameter vector to get the true log likelihood of the physiological data given the PDE parameters, as well as the success labels. Use these points to build a GP emulator and a GP classifier. Only those parameter vectors which yield successful PDE simulations are added to the list of training points for the GP regression model. All points regardless of whether or not they provide a successful simulation are added to the list of training points for the GP classifier, to enable the classifier to learn the infeasible regions that break the biophysical assumptions.
- Exploratory phase. Gather information about the target distribution by running HMC on the surrogate log posterior of the PDE parameters; the end point of every HMC trajectory is subject to a 2-stage DA Metropolis-Hastings accept/reject step (see Equations (15) and (16)), for which the simulator is called. The emulator and the classifier are sequentially refined (i.e., optimum covariance hyperparameters are found by maximisation of the log marginal likelihood of the GP training points – see Equation (A2)) as new points are accepted. Accepted parameter vectors are iteratively added as

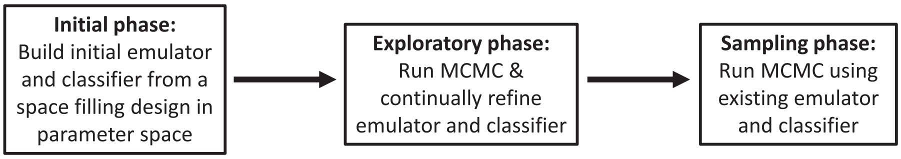
Initial phase:
Build initial emulator and classifier from a space filling design in parameter space
Exploratory phase:
Run MCMC & continually refine emulator and classifier
Sampling phase:
Run MCMC using existing emulator and classifier

FIGURE 2 Workflow of the GPHMC algorithm.[32] The emulator and classifier constructed in the initial phase are continually refined as HMC is run in the exploratory phase. HMC in the sampling phase proceeds by drawing samples from the asymptotically exact posterior distribution, with the use of the emulator and classifier, which are no longer updated

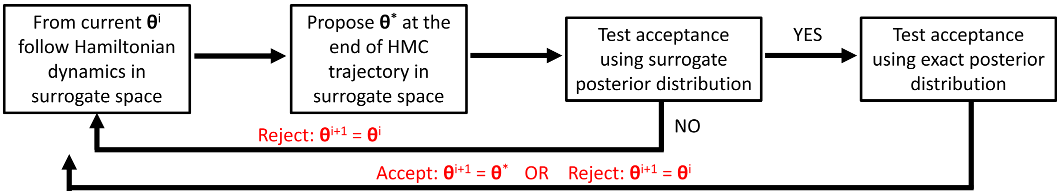
From current θ^i follow Hamiltonian dynamics in surrogate space
Propose θ* at the end of HMC trajectory in surrogate space
Test acceptance using surrogate posterior distribution
YES
Test acceptance using exact posterior distribution
NO
Reject: θ^{i+1} = θ^i
Accept: θ^{i+1} = θ* OR Reject: θ^{i+1} = θ^i

FIGURE 3 Workflow of our proposed Delayed Acceptance within GPHMC[32] method

further training points to those used in the initial design stage. The points in the initial design stage are gradually removed from the list of training points as they tend to come from low posterior density regions and can bias the inference results – see lines 16 and 27 in Algorithm 1b. As HMC explores the parameter space, it gets closer to the equilibrium distribution (burn-in phase). This ensures the algorithm sequentially zooms into the regions of high posterior probability. Following,[32] we set the emulated "potential energy" of the HMC algorithm (see Section 3.2) to

$$\tilde{E}(\boldsymbol{\theta}^*) = \frac{\mathbb{E}(f(\boldsymbol{\theta}^*)|D) - \sqrt{\mathrm{var}(f(\boldsymbol{\theta}^*)|D)}}{2\sigma^2} + \frac{n}{2}\log\left(2\pi\sigma^2\right) - \log\tilde{p}(\boldsymbol{\theta}^*).$$

Here σ^2 is the variance of the errors, $\tilde{p}(\boldsymbol{\theta}^*)$ is the prior distribution (Equation (17)), $f(.)$ is the emulated residual sum-of-squares function, $\mathbb{E}(f(\boldsymbol{\theta}^*)|D)$ is the GP posterior predictive mean given the training points D (see Equation (A4)) and $\sqrt{\mathrm{var}(f(\boldsymbol{\theta}^*)|D)}$ is the GP posterior predictive standard deviation (see Equation (A5)) for the residual sum-of-squares of the physiological data at unseen parameter configurations $\boldsymbol{\theta}^*$ conditional on the training points D. This drives the exploration into regions with high posterior probability (low value of $\mathbb{E}(.)$) or high uncertainty (large value of $\sqrt{\mathrm{var}(.)}$). If $\sqrt{\mathrm{var}(.)}>3$ along the trajectory, the simulation is stopped prematurely before reaching the end of the trajectory, as the algorithm steps into a region of high uncertainty, where the GP needs to be further trained. The log likelihood is computed at this point by numerically solving the PDEs of the biophysical model, and the corresponding success label is obtained. We note, as an aside, that this follows the same exploitation-exploration trade-off principle as in optimistic acquisition functions used for Bayesian optimisation, see sec. IV in Reference 76.

- *Sampling phase*. Use the emulator and the classifier constructed in the exploratory phase to draw samples from the target distribution using HMC, or any of its variants described in Section 3.2. At this stage, the emulator and the classifier are no longer updated. The emulated "potential energy" in the HMC algorithm is set to

$$\tilde{E}(\boldsymbol{\theta}^*) = \frac{\mathbb{E}(f(\boldsymbol{\theta}^*)|D)}{2\sigma^2} + \frac{n}{2}\log\left(2\pi\sigma^2\right) - \log\tilde{p}(\boldsymbol{\theta}^*).$$

Note that the numerator in the first term is the expected sum-of-squares error, which combined with the normalisation term in the middle gives the log likelihood of the data, and the final term is the log prior. The end point of the trajectory is subject to a 2-stage DA Metropolis-Hastings accept/reject step (see Equations (15) and (16)), based on the simulator. The rejection rate is monitored, and this indicates how well the GP emulator has captured the log posterior density. A large number of rejections calls for an extension of the exploratory phase.

The surrogate log likelihood is used throughout the trajectory, and the simulator (true) log likelihood is used in the final accept/reject step, which implies that asymptotically the samples are drawn from the exact target distribution. The accuracy of the emulator can be checked by diagnostics,[81] for example, GP predictions for the log likelihood corresponding to unseen (test) inputs should be close to the true (simulator) log likelihood values, that is, these points should lie on the equality line when plotted against each other. In our analysis, we employ HMC in the exploratory phase. For the sampling phase we employ the algorithms described above: NUTS, Adaptive HMC, Adaptive RMHMC, Adaptive LDMC (the adaptation of the tuning parameters is performed with Bayesian optimisation). Note that in the GPHMC algorithm, when we need to calculate derivatives of the emulated log posterior, we use the fact that GPs are closed under differentiation, that is, the derivatives of a GP are also GPs (though with different covariance structures), provided the kernel is differentiable. See sec. 9.4 in Reference 74 for further details.

5 | SIMULATIONS

5.1 | Software

Simulations are run on a RedHat Enterprise Linux 6 machine with Intel(R) Xeon(R) CPU E5-2680 v2 2.80GHz and 32GB RAM. Our code uses Matlab and implements the GP models using the GPstuff toolbox.[82] To run the No U-turn sampler, Riemann Manifold HMC and Lagrangian Dynamical MC, we used the Matlab implementations developed by the authors of the papers where these algorithms were proposed, and they are available as follows.[62-64] The simulated pressure waveforms are obtained by numerically integrating the PDEs described in Section 2.1 with a two step Lax-Wendroff Scheme[83] implemented in C++ by Olufsen et al.[6] The numerical integration of the PDEs to obtain one single model solution requires 1 s CPU time and 24 s elapsed time on our hardware.

Our software can be found at: https://github.com/LMihaelaP/BayesianUQ_CardiovascularModelling.git.

5.2 | Method implementation details

5.2.1 | GP compact support

The emulator needs compact support. The bounds for the parameters were taken to be biologically meaningful, as indicated in Table 1.

5.2.2 | GP models

In our analysis, for GP regression, that is, for building the emulator, we use a squared exponential kernel (see sec. 4.2 in Reference 74) for the covariance function, as chosen by cross-validation based on the data. Additionally, for the GP classifier a Matérn 3/2 kernel (see sec. 4.2 in Reference 74) is used, which captures the classification boundaries, which are assumed not very smooth.** We approximate the intractable integral in Equation (A12), which is required for obtaining the class probabilities, with expectation propagation; see sec. 3.6 in Reference 74 for details.

5.2.3 | GP analytical derivatives

For HMC we require the first order derivatives of the GP posterior predictive mean and variance – see Equations (A6) and (A7), while for RMHMC and LDMC we need the second and third order derivatives, which we obtain analytically,†† since differentiation is a linear operator, so the derivative of a GP is again a GP for differentiable kernels; see sec. 9.4 in Reference 74.

5.2.4 | GPHMC phases

Initial design stage. The first design stage of GPHMC comprises 600 PDE evaluations for parameter configurations generated from a Sobol sequence, which helps us build the initial GP emulator and classifier. The number of PDE evaluations is chosen to ensure a certain accuracy of the emulator, determined by the efficiency of the MCMC sampler in the beginning of the exploratory phase (the acceptance rate of the sampler running on the initial emulator should not be too low (e.g., below 20%).

Exploratory phase. In the exploratory phase of GPHMC[32] we run HMC for 1000 iterations. We tried to ensure that a minimum number of training points is stored (as to boost computational efficiency), while preserving a high enough emulator accuracy (as quantified by GP diagnostics[81]), and a high acceptance rate in the sampling phase (>65%[22]). The samples collected in the exploratory phase are used to refine the GP models for regression and classification (i.e., update the covariance hyperparameters by re-optimising the log marginal likelihood of the GP training points – see Equation (A2)) and to get closer to the equilibrium distribution. The emulator to be used in the sampling phase consists of 400 training points.

Sampling phase. We apply the GPHMC scheme described in Section 4, with the following algorithms in the sampling phase:

- No U-turn sampler algorithm,
- Adaptive Hamiltonian Monte Carlo (AHMC) with Bayesian optimisation for performance tuning,
- Adaptive Riemann Manifold Hamiltonian Monte Carlo (ARMHMC) with Bayesian optimisation for performance tuning,
- Adaptive Lagrangian Dynamical Monte Carlo (ALDMC) algorithm with Bayesian optimisation for performance tuning.

The performance tuning involves setting the integration step size and the number of integration steps between subsequent Metroplis-Hastings acceptance steps.

In the sampling phase of GPHMC[32] we allow for further 500 samples as burn-in phase, and 5000 samples are subsequently drawn and used for inference. The chosen number of burn-in steps ensures that the value for the multivariate potential scale reduction factor (MPSRF) falls below the standard threshold of 1.1, MPSRF $\leq$ 1.1,[84] which is taken as an indication of sufficient convergence.

5.2.5 | Bayesian optimisation for HMC performance tuning

The adaptive algorithms with Bayesian optimisation[65] require us to construct an initial GP for the normalised squared jumping distance in Equation (12), as a function of the step size and the number of leapfrog steps, which need compact support. For the adaptive algorithms, we use the following ranges for the step size and the number of leapfrog steps: for HMC, $L \in \{1, ..., 50\}$, $\epsilon \in [10^{-4}, \ 5 \times 10^{-3}]$, and for RMHMC and LDMC, $L \in \{1, ..., 40\}$, $\epsilon \in [10^{-2}, \ 8 \times 10^{-2}]$. These ranges are chosen in a way that ensures a high effective sample size (ESS),[85] while no anticorrelation is induced (which would make ESS greater than the total number of MCMC samples).[86] The initial GP for the normalised squared jumping distance is constructed based on 20 (ϵ, L) parameter vectors, and for each parameter vector we obtain 10 MCMC samples for the PDE parameters to estimate the expectation in Equation (12).

5.2.6 | MCMC convergence with MPSRF

Every algorithm in the sampling phase is run 10 times from different random number generator seeds and different starting values for the parameters, selected from the points collected in the exploratory phase, to make it less likely that we start in a low probability region. In choosing the number of independent chains, we followed the suggestion given in the study by Cowles et al.[87] The authors advise to use 10 independent chains if the posterior distribution is unimodal, and more for multi-modal distributions. Generally, as proposed in the study by Gelman and Rubin,[88] an optimisation, mode-finding algorithm may be used to find regions of high density, and the starting values can be generated by sampling from a mixture of t-distributions centred at these modes.

5.2.7 | MCMC mixing with ESS

For each of the 10 simulations, we record the minimum ESS across all parameters (i.e., compute ESS for each of the four parameters, and take the minimum from this set of 4, see Equation (20)), the minimum ESS divided by the number of forward (PDE) simulations, and the minimum ESS divided by the total CPU time for the entire simulation.

5.2.8 | MCMC set-up

We apply all the algorithms to the pulmonary circulation model, to carry out a comparative assessment of their performance. The data used (pulmonary blood pressure) are described in Sections 2.3 and 2.4. The four biophysical parameters that are to be learned, given in Equation (5), are the vessel stiffness and the Windkessel parameters.

We assume the following:

- Likelihood based on the simulator: $y_i \,|\, \boldsymbol{\theta} \sim \mathcal{N}(m_i(\boldsymbol{\theta}), \sigma^2)$, that is,

$$p(\mathbf{y}|\boldsymbol{\theta},\sigma^2) = \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^n \exp\left(-\frac{\sum\limits_{i=1}^{n}(y_i - m_i(\boldsymbol{\theta}))^2}{2\sigma^2}\right), \tag{18}$$

where $\mathbf{m}(\boldsymbol{\theta}) = (m_1(\boldsymbol{\theta}), ..., m_n(\boldsymbol{\theta}))$ is the vector of predictions from the PDEs of Equations (1)–(3), $\mathbf{y} = (y_1, ..., y_n)$ is the vector of measured data (pressure data), and n is the number of time points. If a PDE simulation is unsuccessful, we set the likelihood to zero in the MCMC simulation, that is, $p(\mathbf{y}|\, \boldsymbol{\theta}, \sigma^2) = 0$.

- Likelihood based on the emulator:

$$\tilde{p}(\mathbf{y}|\boldsymbol{\theta},\sigma^2) = \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^n \exp\left(-\frac{\tilde{\mathcal{S}}(\boldsymbol{\theta})}{2\sigma^2}\right), \tag{19}$$

where $\tilde{\mathcal{S}}(\boldsymbol{\theta})$ is the value of the residual sum-of-squares $\sum_{i=1}^{n}(y_i - m_i(\boldsymbol{\theta}))^2$ predicted by the emulator (the GP posterior predictive mean – Equation (A4)) for the particular $\boldsymbol{\theta}$. The covariance hyperparameters are refined by optimisation of the log marginal likelihood for the GP regression (see Equation (A2)) and classification (see Equation (A12)) at every step in the exploratory phase of the GPHMC algorithm.[32] The emulator and classifier thus created are used in the sampling phase, with no further updates.

- Priors: We use a rescaled Beta(1, 1) distribution (to ensure positive support for the biophysical parameters) with support within physiologically realistic ranges (given in Table 1), $\theta_i \sim$ Beta(1, 1), $\quad l_i \le \theta_i \le u_i$, where $i = 1, ..., d$, with $d = 4$ being the parameter dimensionality. For the emulation phase, this prior is modified by the GP classifier according to Equation (17).

For the noise variance σ^2, we follow our previous work in Reference 7, and use a conjugate weakly informative inverse gamma prior, $\sigma^2 \sim$ Inv - Gamma(a, b), with $a = 0.5n_s$ and $b = 0.5n_s\gamma_s^2$, where $\gamma_s^2 = \frac{\min\left(\sum_{i=1}^{n}(y_i - m_i(\boldsymbol{\theta}))^2\right)}{n-d}$ is the prior value for σ^2 and $n_s = 1$ is the prior accuracy for γ_s^2:

$$p(\sigma^2) = \frac{b^a}{\Gamma(a)}(\sigma^2)^{-a-1}\exp\left(-\frac{b}{\sigma^2}\right).$$

- Posterior distribution:

$$p(\boldsymbol{\theta},\sigma^2|\mathbf{y}) \propto p(\mathbf{y}|\boldsymbol{\theta},\sigma^2)p(\boldsymbol{\theta})p(\sigma^2),$$

and

$$\tilde{p}(\boldsymbol{\theta},\sigma^2|\mathbf{y}) \propto \tilde{p}(\mathbf{y}|\boldsymbol{\theta},\sigma^2)\tilde{p}(\boldsymbol{\theta})p(\sigma^2).$$

The parameters $\boldsymbol{\theta}$ are sampled using all the algorithms tested. At every MCMC iteration, given the value for $\boldsymbol{\theta}$, σ^2 is sampled with a Gibbs sampling step from $p(\sigma^2|\, \boldsymbol{\theta}, \mathbf{y})$, which due to the conjugacy of the prior is available in closed-form:

$$p\left(\sigma^2|\boldsymbol{\theta},\mathbf{y}\right)=\text{Inv-Gamma}\left(\frac{n}{2}+a,0.5\sum_{i=1}^{n}(y_i-m_i(\boldsymbol{\theta}))^2+b\right).$$

We keep σ^2 fixed in the burn-in phase and infer it in the sampling phase.

5.2.9 | Parameter transformations

To improve numerical stability and reduce round-off errors,[7] the original parameters θ_i are scaled in the order of 1 for building the GP emulator and GP classifier: $\theta_i \in [l_i,u_i] \rightarrow \frac{\theta_i}{s_i} \in [-1,1]$, where s_i is a scaling factor which ensures $\frac{\theta_i}{s_i} \in [-1,1]$. In addition, the HMC algorithm requires unbounded parameters: $\theta_i \in [l_i,u_i] \rightarrow \log_e \frac{\theta_i - l_i}{u_i - \theta_i} \in \text{IR}$. The transformed parameters are then mapped back via the inverse transformation into the original domain $\theta_i \in [l_i, u_i]$ for the PDE simulator, that is, when they are inserted into the mathematical PDE model described in Section 2.1.

6 | NUMERICAL RESULTS

6.1 | Physiological data

6.1.1 | Accuracy

In Figure 4 (bottom panel) we superimpose kernel density estimation plots for the marginal posterior densities of the PDE parameters $\boldsymbol{\theta} = (s, r_1, r_2, c)$, obtained with the various emulation HMC algorithms compared in our study. To test whether the emulation approach gives any bias in the results, we also present results obtained with a long-run MCMC sampler (the Adaptive Metropolis [AM] algorithm[89]) that draws samples directly from the asymptotically exact posterior distribution, which we take as a proxy for the gold standard. HMC running directly on the original posterior distribution would incur excessive computational costs (see Section 4 for a discussion on this), hence in our work we opt for a random-walk algorithm (AM). To obtain the marginal posterior densities, we have used the kernel smoothing function estimate for univariate data with the optimal bandwidth for normal densities.[90] We have also constructed QQ plots (see the top panel of Figure 4), where we plot the marginal posterior densities from the direct AM algorithm against any emulation HMC algorithm for the four parameters. The kernel density plots show overlapping densities for the different algorithms, and the QQ plot suggests that the algorithms provide samples from approximately the same density. In addition, in Figure A1 of the Appendix we superimpose the pairwise scatterplots for all methods: GP AHMC, GP NUTS, GP ARMHMC and GP ALDMC. We notice great overlap between the pairwise points for all algorithms, indicating agreement between the methods in input space. Therefore, given that all the densities predicted by the different sampling algorithms tend to agree and are very similar, and there is no indication of lack of convergence (as seen from Table 3), quantified using MPSRF[84] (MPSRF $\leq$1.1), we have strong evidence that all sampling algorithms included in our study converge to the true posterior distribution, and no bias is introduced by the emulator.

6.1.2 | Model fits

Next, we inspect the parameter estimation results. We have simulated the posterior median blood pressure time series (50% quantile) obtained from the simulations with the GP ALDMC sampler. We have superimposed the measured pressure waveform from the laboratory, obtained with right-heart catheterisation. In addition, we have plotted the 95% predictive credible interval, calculated analytically as

$$m(\boldsymbol{\theta}) \pm 1.96\sigma_{\text{total}},$$

where $m(\boldsymbol{\theta})$ is the mean posterior pressure signal, and

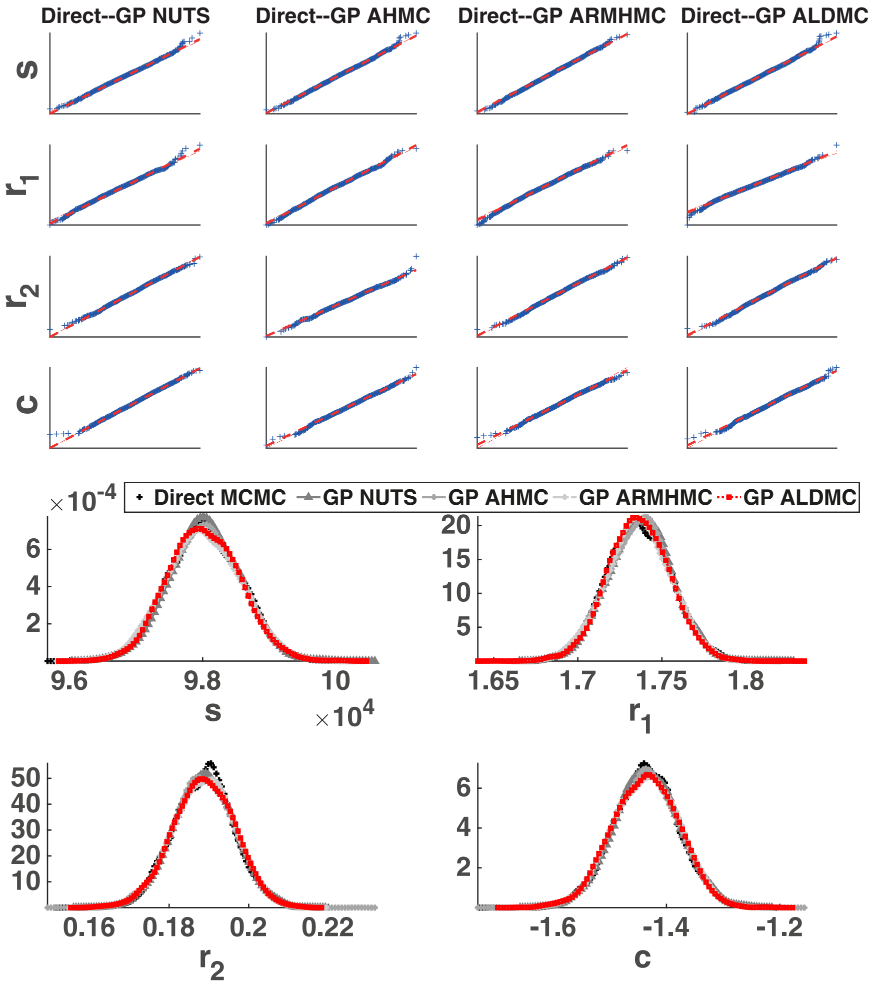

FIGURE 4 Measured (physiological) data inference results. Top panel: QQ plots, where we plot the marginal posterior densities obtained with every emulation algorithm against the marginal posterior densities obtained with the long MCMC reference run based on the actual PDE model rather than the emulator for the four parameters (s: stiffness, r_1, r_2, c: Windkessel parameters, see Equation (5)). Bottom panel: Kernel density estimation plots for the marginal posterior probability densities of the four parameters (s: stiffness, r_1, r_2, c: Windkessel parameters, see Equation (5)) for the GP emulation algorithms and the direct MCMC algorithm. To obtain the marginal posterior density, we have used the kernel smoothing function estimate for univariate data with the optimal bandwidth for normal densities.[90] Samples were drawn using the MCMC algorithms from Section 3.2, which were combined with the GP emulator and classifier as described in Section 4. The Adaptive Metropolis[89] algorithm was used to draw samples approximately from the exact posterior distribution. This enables to test if the emulation approach gives any bias. The data used for this study are measured (physiological) pulmonary pressure time series described in Section 2.3, for which the true parameter values are unknown. Each simulation for the emulation algorithms consisted of 5000 sampling phase draws and 500 burn-in draws (the latter were discarded), while for the direct MCMC algorithm, the simulation consisted of 150,000 sampling phase draws and 500 burn-in draws, which were discarded. Legend: GP: Gaussian Process; NUTS: No U-turn Sampler; AHMC: Adaptive Hamiltonian Monte Carlo; ARMHMC: Adaptive Riemann Manifold Hamiltonian Monte Carlo; ALDMC: Adaptive Lagrangian Dynamical Monte Carlo

$$\sigma_{\text{total}} = \sqrt{\sigma^2_{\text{error}} + \sigma^2_{\text{signal}}},$$

with σ^2_{error} being the mean posterior estimate for the noise variance, and σ^2_{signal} being the variance of the distribution of the signals, which are model predictions obtained with the MCMC posterior parameter samples. The results are shown in the top panel of Figure 5. We notice that the 95% CI contains most of the measured data and that the measured and generated waveforms are in reasonable agreement, except for the peak, where a mismatch is registered. This points to

TABLE 3 A comparison of the various MCMC algorithms, reviewed in Section 3.2, tested on the measured (physiological) pulmonary blood pressure time series described in Section 2.3

Algorithm	Pre-processing: CPU time (s) and no. PDEs	Sampling: CPU time (s) and no. PDEs	$\frac{\text{Min(ESS)}}{\text{N}}$	$\frac{\text{Min(ESS)}}{\text{CPUtime}}$	$\frac{\text{Min(ESS)}}{\text{no.PDEs}}$	Acceptance probability	MPSRF
GP NUTS	0 and 0	6025 (87) and 4841 (10)	0.21 (0.02)	0.18 (0.01)	0.22 (0.02)	0.97 (0.002)	1.001
GP AHMC	539 and 198	8908 (362) and 4929 (11)	0.29 (0.05)	0.16 (0.02)	0.30 (0.05)	0.98 (0.003)	1.001
GP ARMHMC	2102 and 199	35,390 (1675) and 4997 (2)	0.61 (0.12)	0.09 (0.02)	0.60 (0.12)	0.99 (0.001)	1.001
GP ALDMC	499 and 200	7570 (509) and 4996 (2)	0.65 (0.14)	0.43 (0.08)	0.65 (0.14)	0.99 (0.001)	1.000

Note: These were carried out in the sampling phase of the GPHMC algorithm described in Section 4. Ten chains with 5000 sampling phase draws and 500 burn-in draws (which were discarded) were run. The mean and the standard deviation in brackets is provided for the minimum effective sample size normalised by the total number of MCMC samples N, across all four parameters (see Equation (20)), and the mean is obtained over all 10 simulations. We also provide min(ESS) normalised by CPU time, min(ESS) normalised by the number of forward (PDE) evaluations, the acceptance probability, the multivariate potential scale reduction factor (MPSRF), as well as the pre-processing CPU times and number of forward evaluations, and the sampling phase CPU times and number of forward evaluations. For the adaptive algorithms, 200 MCMC samples are drawn and used for pre-processing, needed for building an initial GP for the normalised squared jumping distance, which is used in the Bayesian optimisation; NUTS does not require any pre-processing. Legend: GP: Gaussian Process; NUTS: No U-turn Sampler; AHMC: Adaptive Hamiltonian Monte Carlo; ARMHMC: Adaptive Riemann Manifold Hamiltonian Monte Carlo; ALDMC: Adaptive Lagrangian Dynamical Monte Carlo.

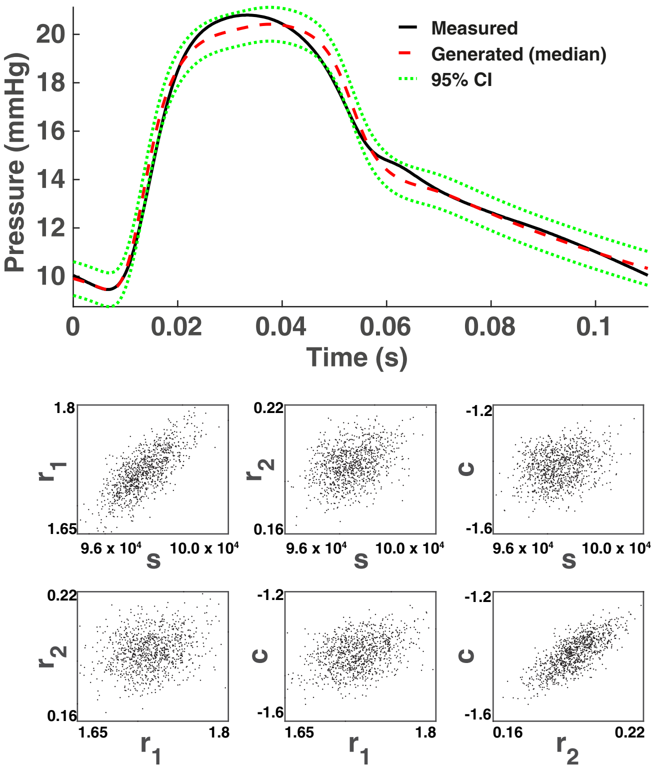

FIGURE 5 Top panel: Measured (physiological) blood pressure time series (black solid line) compared with simulated pressure time series (red dashed line), which is the posterior median signal obtained from the simulations with the Gaussian Process Adaptive Lagrangian Dynamical Monte Carlo sampler (GP ALDMC). With green dotted lines we show the 95% predictive credible interval, calculated analytically as $m(\boldsymbol{\theta}) \pm 1.96\sigma_{\text{total}}$, where $m(\boldsymbol{\theta})$ is the mean posterior pressure signal, and $\sigma_{\text{total}} = \sqrt{\sigma^2_{\text{error}} + \sigma^2_{\text{signal}}}$, with σ^2_{error} being the mean posterior estimate for the noise variance, and σ^2_{signal} being the variance of the distribution of the signals, which are model predictions obtained with the MCMC posterior parameter samples. Bottom panel: Parameter correlations obtained from the posterior samples based on the measured data

the existence of a slight model mismatch, and in Section 7 we offer a potential explanation. In addition, in Figure A2 of the Appendix we include the posterior median pressure signal and the 95% CI for all methods: GP AHMC, GP NUTS, GP ARMHMC and GP ALDMC. We notice substantial overlap beyond the figure's resolution level between the medians and the CIs, indicating that all four algorithms provide very similar model predictions and UQ. This indicates agreement between the methods in output space.

6.1.3 | Posterior correlations

We have also investigated the existence of posterior correlations between the parameters (bottom panel of Figure 5) by plotting the pairwise scatterplots between the parameters. It appears that there is a moderate to strong positive correlation between the stiffness parameter, s, and the first resistance adjustment parameters, r_1, as well as between the second resistance adjustment parameters, r_2, and the compliance adjustment parameter, c.

6.1.4 | Efficiency

For each of the emulation algorithms described in Section 3.2, we have run 10 simulations with different starting values for the parameters and different random number generator seeds. Table 3 summarises the simulation CPU time and the number of forward simulations (numerical PDE integrations) for every algorithm, the acceptance rate, the efficiency quantified using ESS, and the convergence in terms of MPSRF. Figure 6 also provides a comparison between these algorithms by displaying the distribution over 10 simulations of the minimum ESS normalised by the total number of MCMC samples N, and the minimum ESS is calculated by computing ESS for each of the four parameters, and taking the minimum from this set of 4:

$$\text{ESS}^* = \min_k \text{ESS}_{ik}, \tag{20}$$

where $i = 1, ..., 10$ represents the simulation and $k = 1, ..., 4$ the parameter label. We also show the minimum ESS in Equation (20) normalised by CPU time (i.e., the distribution based on 10 simulations of the minimum ESS normalised by the CPU time of each simulation), and the minimum ESS in Equation (20) normalised by the number of PDE evaluations for each of the 10 simulations. We note here that ESS[85] estimates the effective number of independent samples out of the total number of MCMC samples, and a low value can indicate inefficiency of the sampler.

CPU times. All algorithms share the initial and the exploratory phase of the GPHMC algorithm. We thus start to compare the different algorithms in the sampling phase. The adaptive algorithms require pre-processing, needed by the Bayesian optimisation scheme, while NUTS does not (Table 3). AHMC records somewhat higher CPU time than ALDMC in the pre-processing and the sampling phase, despite the number of PDE evaluations required by each algorithm being somewhat similar (Table 3). This can be explained by the fact that the AHMC algorithm makes smaller steps and needs a larger number of steps for an efficient exploration of the parameter space when compared to ALDMC.‡‡ In addition, the CPU time is much larger for ARMHMC compared to ALDMC, despite the number of PDEs evaluated being the same. An explanation for this is the high computational costs of the numerical scheme utilised by ARMHMC to solve the Hamiltonian equations (see Section 3.2 for more details on this).

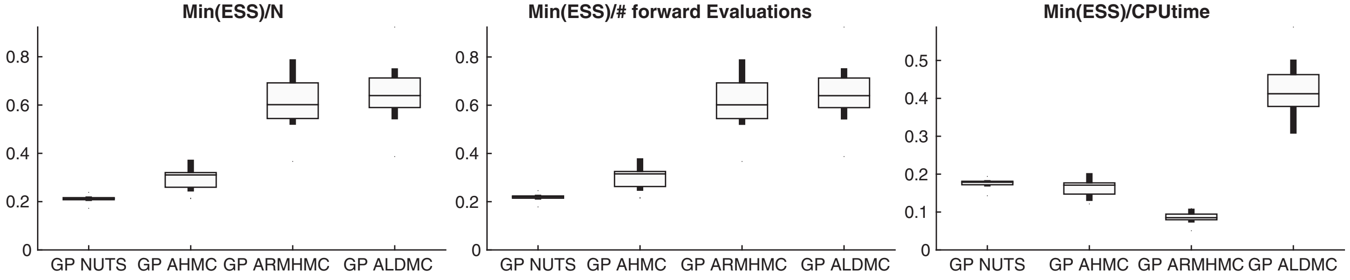

FIGURE 6 Measured (physiological) data inference results. Left panel: Distribution over 10 simulations of minimum Effective Sample Size (ESS) – Equation (20) – normalised by the total number of MCMC samples N. Middle panel: min(ESS) in Equation (20) normalised by the number of forward (PDE) evaluations. Right panel: min(ESS) in Equation (20) normalised by CPU time, for all algorithms tested, as part of the sampling phase in the GPHMC algorithm described in Section 4 for the measured (physiological) pulmonary blood pressure time series described in Section 2.3. Distribution obtained based on 10 chains, with 5000 sampling phase draws and 500 burn-in draws, where the latter were discarded. Legend: NUTS: No U-turn Sampler; AHMC: Adaptive Hamiltonian Monte Carlo; ARMHMC: Adaptive Riemann Manifold Hamiltonian Monte Carlo; ALDMC: Adaptive Lagrangian Dynamical Monte Carlo

ESS. When looking at the minimum ESS, ESS* (Equation (20)) normalised by the total number of MCMC samples N (left panel of Figure 6), we observe that the two algorithms which use gradient and curvature information from the log posterior distribution, namely ARMHMC and ALDMC, perform better than the algorithms which only use gradient information, namely NUTS and AHMC. It is clear that allowing the "mass matrix" $\mathbf{M}$ to adapt to the curvature of the log posterior distribution is highly beneficial. We note that the performance of ALDMC and ARMHMC is comparable, while AHMC registers a higher ESS than NUTS. A very similar trend is observed when analysing the minimum ESS in Equation (20) normalised by the number of PDE evaluations (middle panel of Figure 6). However, when the CPU time is taken into account (right panel of Figure 6), the ALDMC algorithm scores highest. In addition, ARMHMC pays the price of high computational costs, and is the least efficient algorithm in terms of CPU time. NUTS and AHMC have a similar performance, the advantage of NUTS is that no pre-processing is required. In our case, the Bayesian optimisation scheme is not more efficient than the scheme employed by NUTS for the tuning of L and ϵ (when ESS normalised by CPU time is considered).

6.2 | Synthetic data

We re-run the analysis on synthetic data generated as described in Section 2.4 to investigate if the differences between methods hold for both synthetic and physiological data equally. The rationale for this comparison is to investigate how the model mismatch, which is only present in the real data but absent in the synthetic data, affects the inference and the performance of our methods.

6.2.1 | Accuracy

Figure 7 shows the marginal posterior densities obtained with kernel density estimation from the posterior samples for all four emulation methods. We superimpose the true parameter values which generated the synthetic data in a black vertical line. The marginal posterior densities generated with the different methods overlap, and the true parameter values are contained within the marginal posterior densities for all four parameters, which is consistent with our inference procedure. In the Appendix A.6, we offer an explanation as to why the posterior densities are not centred at the true parameter values.

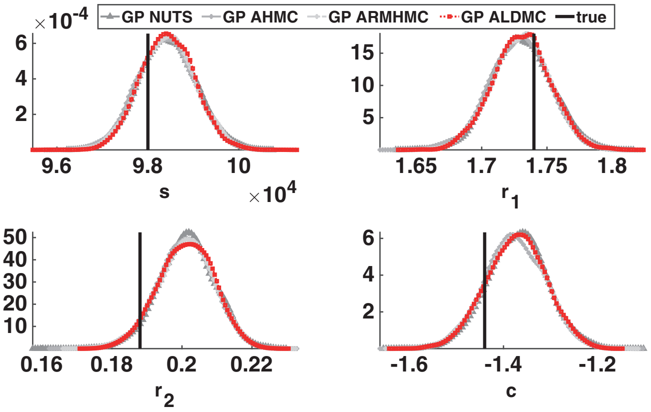

FIGURE 7 Synthetic data inference results. Kernel density estimation plots for the marginal posterior probability densities of the four parameters (s: stiffness, r_1, r_2, c: Windkessel parameters, see Equation (5)) for the GP emulation algorithms. To obtain the marginal posterior density, we have used the kernel smoothing function estimate for univariate data with the optimal bandwidth for normal densities.[90] Samples were drawn using the MCMC algorithms from Section 3.2, which were combined with the GP emulator and classifier as described in Section 4. The data used are synthetic pulmonary pressure time series, generated as described in Section 2.4. The true parameter values which generated these data are superimposed as a black vertical line. Each simulation for the emulation algorithms consisted of 5000 sampling phase draws and 500 burn-in draws (the latter were discarded). Legend: GP: Gaussian Process; NUTS: No U-turn Sampler; AHMC: Adaptive Hamiltonian Monte Carlo; ARMHMC: Adaptive Riemann Manifold Hamiltonian Monte Carlo; ALDMC: Adaptive Lagrangian Dynamical Monte Carlo

6.2.2 | Efficiency

Figure 8 displays the distribution over 10 simulations of minimum ESS normalised by *N*, by the number of forward evaluations and by the CPU time. In terms of ESS normalised by *N* and number of model solutions, similar observations as for the physiological data can be made. NUTS remains the algorithm with the worst performance, AHMC is second worst and ARMHMC has a fairly similar performance to ALDMC. When analysing ESS/CPUtime, ALDMC remains the best algorithm, however, a larger discrepancy between AHMC and NUTS can be observed, with NUTS being clearly the second best. Interestingly, AHMC performs equally as poorly as ARMHMC.

7 | DISCUSSION

7.1 | Methodological approach

In the present study, we have combined GPs with state-of-the-art MCMC schemes to enable parameter estimation and uncertainty quantification in an analytically non-tractable, non-linear system of coupled PDEs at manageable computational costs. This scheme differs from the approach of gradient matching, proposed for example, in Reference 91, where a GP is used to approximate the solutions of the differential equations and thereby avoid the computationally expensive numerical integration (see[92] for the potential limitations). In our approach, there is no need to approximate the solution of the differential equations itself. Instead, we apply GP regression to approximate the likelihood function, with the objective to reduce the number of PDE evaluations required to calculate the likelihood during the MCMC iterations. We have evaluated various variants of this paradigm in terms of accuracy and efficiency on a biophysical fluid-dynamics model of the pulmonary circulation. We have also built a GP classifier to automatically learn the regions in the parameter space that violate the physical assumptions of the mathematical model.

Our study presents several new methodologies. We have extended the GPHMC algorithm[32] to allow for the use of other HMC variants (e.g., RMHMC), and we have coupled this with Bayesian optimisation for automatic parameter tuning. In addition, the combination of HMC and Delayed Acceptance[53,55] is novel. In terms of the application of our work, we have used fast Bayesian sampling methods to quantify uncertainty in the topical research area of quantitative physiology and pathophysiology, with a specific focus on the pulmonary circulation system, which is potentially interesting to clinical practitioners for detection of long-term pulmonary hypertension. To our knowledge, employing an MCMC scheme and coupling it with GPs to emulate the likelihood, while performing approximate exact (meaning asymptotically converging to the true posterior probability distribution) inference has only been performed in one other study[61] in the cardiovascular mathematical modelling field before, but with no extensive algorithm comparison. Most studies rely on parameter optimisation to estimate parameters,[3-6] which only provides a point estimate (i.e., a specific value) for the parameters, without allowing for an understanding of the uncertainty in these values. In addition, obtaining posterior

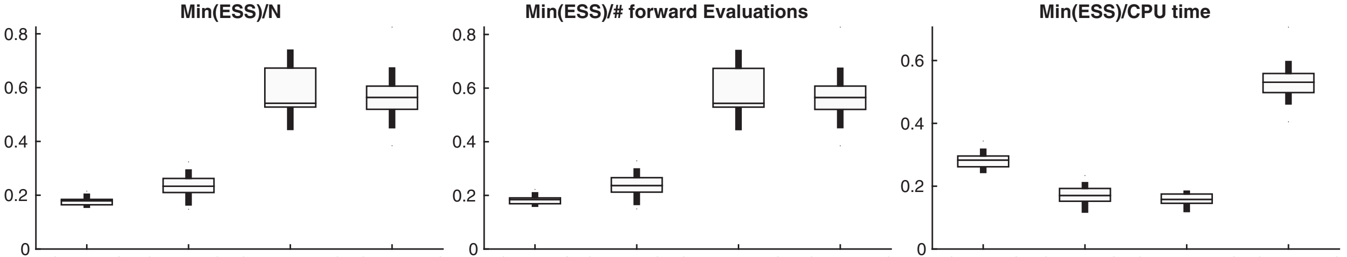

FIGURE 8 Synthetic data inference results. Left panel: Distribution over 10 simulations of minimum Effective Sample Size (ESS) – Equation (20) – normalised by the total number of MCMC samples *N*. Centre panel: min(ESS) in Equation (20) normalised by the number of forward (PDE) evaluations. Right panel: min(ESS) in Equation (20) normalised by CPU time, for all algorithms tested, as part of the sampling phase in the GPHMC algorithm described in Section 4 for the synthetic pulmonary blood pressure time series generated as described in Section 2.4. Distribution obtained based on 10 chains, with 5000 sampling phase draws and 500 burn-in draws, where the latter were discarded. Legend: NUTS: No U-turn Sampler; AHMC: Adaptive Hamiltonian Monte Carlo; ARMHMC: Adaptive Riemann Manifold Hamiltonian Monte Carlo; ALDMC: Adaptive Lagrangian Dynamical Monte Carlo

distributions for the parameters allows an in-depth understanding of the correlations between parameters, or construction and interpretation of credible intervals, which can reveal if the parameters can be used as disease indicators in the clinic.

7.2 | Algorithm accuracy

Hamiltonian and Lagrangian algorithms (NUTS, AHMC, ARMHMC, ALDMC) have been employed, and they have all provided parameter posterior samples from approximately the same distribution, both for the physiological data study and synthetic data study (Figures 4 and 7 and Figure A1). The synthetic study results indicate that the true parameter values that generated the data are contained within the marginal posterior densities (Figure 7), which is consistent with our inference procedure. Additionally, the model predictions are very similar, all closely following the measured (physiological) data, and the UQ in output space is comparable (Figure A2) between algorithms. For the real data, for which the true parameter values are unknown, the agreement between the posterior probability densities in input space (Figure 4), and between model predictions and UQ in output space (Figure A2) across all algorithms is taken as a proxy for accuracy.

7.3 | Posterior correlations

The existence of moderate to strong posterior correlations can be observed from Figure 5 (bottom panel). Analysing posterior correlations can reveal whether the model parameters can be uniquely identified from the data. Such insight into posterior parameter estimation uncertainty and correlation is important when considering whether the estimated parameter values are informative disease indicators to be used in a clinical decision support tool. However, such a tool needs to work in real time, and computational efficiency, which we discuss next, is therefore paramount.

7.4 | Algorithm efficiency

7.4.1 | MPSRF versus ESS

The advantage of NUTS over the other algorithms is that it registers MPSRF < 1.1 while requiring zero pre-processing effort. In addition, NUTS records the lowest CPU time in the sampling phase due to the lowest number of PDEs evaluated, while maintaining a large acceptance rate (97%). However, NUTS registers the poorest mixing and thus, lowest (worst) efficiency, as quantified by min(ESS)/N, which is the case for both the physiological and synthetic data studies. ESS is a measure of the algorithm's mixing, thus efficiency, by capturing the amount of information that the correlated MCMC samples contain about the target density, that is, ESS gives the equivalent number of independent samples out of the total number of MCMC correlated samples N. If the sampler exhibits poor mixing, it indicates that the Markov chain moves slowly through the parameter space, leading to low ESS relative to N. In that case, the samples drawn are highly correlated, and the sampler needs to be run for longer in order to acquire a sufficiently large number of independent samples and ensure a thorough exploration of the parameter space. In contrast, MPSRF indicates whether there are any signs of non-convergence to the target density of the MCMC sampler. It is possible that the sampler appears to have converged to the target density, that is, MPSR<1.1, however the sampler's mixing, quantified via ESS, is poor, with high correlations between the MCMC samples, see our previous study[7] for an example.

7.4.2 | ESS normalised by *N*

The algorithms which make use of curvature information from the log posterior distribution (ARMHMC, ALDMC) achieved higher efficiency (in terms of min(ESS)/N) than the algorithms which only use gradient information (AHMC, NUTS) for both the physiological and synthetic data studies. In applications from literature studies,[63-65] and for the application presented here, (A)LDMC and (A)RMHMC use a larger step size and a smaller number of steps compared to (A)HMC and NUTS. The reason is that (A)LDMC and (A)RMHMC set the mass matrix based on the curvature of the log posterior, while (A)HMC and NUTS use an identity matrix as the mass matrix. The implication is that for (A)HMC and NUTS all the work falls on the back of the step size (and number of steps) to optimise efficiency, and the optimum step size is restricted by the lowest marginal variance. In contrast, for (A)LDMC and (A)RMHMC this issue is alleviated by the use of a non-identity mass matrix. It is expected that the difference between (A)HMC/NUTS and (A)LDMC/(A) RMHMC is less pronounced for problems with similar marginal variances, for which the optimum step size would be similar across different parameters (see sec. 10.1.1 in Reference 63 for an example). The distance in space, $\epsilon \times L$, travelled by (A)LDMC and (A)RMHMC is generally larger than for (A)HMC and NUTS, leading to enhanced exploration efficiency of the former two compared to the latter two algorithms. This explains why min(ESS)/N is larger for ALDMC and ARMHMC compared to AHMC and NUTS. ALDMC and ARMHMC register fairly similar efficiency in terms of min(ESS)/N, and AHMC registers higher min(ESS)/N than NUTS for both data sets: physiological and synthetic. A potential explanation for the latter finding is that for NUTS the tuning of ϵ and L is performed in the emulated log posterior entirely, based on parameter samples accepted at the emulator stage, due to the construction of the algorithm, while for AHMC the tuning using Bayesian optimisation is based on samples accepted at the simulator stage, thus the simulator plays a role in finding the optimum tuning parameters, positively impacting efficiency. The finding that AHMC coupled with emulation has superior efficiency to NUTS with emulation is in line with the findings in Reference 65 comparing the conventional AHMC (i.e., without emulation) with the conventional NUTS.

7.4.3 | ESS normalised by CPU time

The efficiency and the acceptance rate of AHMC are enhanced at the cost of a larger number of PDEs being evaluated in the sampling phase, which results in larger CPU times compared to NUTS, thus in terms of min(ESS) normalised by CPU time, AHMC loses its advantage over NUTS for both the physiological and synthetic data studies. When comparing ALDMC to AHMC, the following can be noted. AHMC has the advantage of avoiding calculations of higher-order derivatives, but requires a larger number of steps to be made, while ALDMC calculates higher-order derivatives, but needs fewer steps. This appears to result into somewhat lower CPU times for ALDMC (Table 3), which coupled with a larger min(ESS), leads to ALDMC having a clear advantage in the case of physiological and synthetic data. Additionally, for both data sets, the ARMHMC algorithm loses its computational advantage due to the computationally burdensome integration scheme for the Hamiltonian dynamics (see Section 3.2), and scores the lowest. What is more, despite NUTS requiring a lower CPU time to run, ALDMC registers a min(ESS)/N nearly three times larger than NUTS, making the latter algorithm computationally most efficient in terms of ESS/CPU time.

7.4.4 | ESS normalised by number of forward evaluations

Regarding min(ESS) normalised by the number of forward evaluations, the same pattern can be observed as for min(ESS)/N, since the number of forward solutions is fairly similar to N.

Different dynamical systems are based on differential equations with different numerical integration costs. A computational complexity measure that is normalised by the number of forward simulations can be generalised across different dynamical systems, whereas other measures, like CPU time, cannot (as the differences in computational cost may merely reflect the differences in the numerical complexity of the underlying differential equations). Thus, the metric based on ESS/number of forward solutions can be used to compare strictly the performance of the algorithms, unlike ESS/CPU time. The comparative observations made about (A)HMC, NUTS, (A)LDMC and (A)RMHMC are based on empirical findings, to our knowledge no theoretical study comparing the efficiency between the algorithms has been performed.

7.4.5 | Real-time treatment planning

Table 4 shows that the number of forward solutions per HMC iteration avoided by the coupling with a GP surrogate model is more than 100 on average. Therefore, a reduction in the number of forward solutions by two orders of magnitude is obtained. This demonstrates that using efficient surrogate models in place of the computationally expensive PDE model is key in assisting clinicians in real-time treatment planning.

TABLE 4 Number of model evaluations (PDEs) required to obtain one single HMC sample drawn using conventional HMC versus emulation HMC (GPHMC) algorithm (mean and standard deviation in brackets)

Algorithm	Number of PDEs per HMC sample
Conventional HMC	117 (65)
Emulation HMC	1

Note: The number of PDEs for the conventional HMC is $L(d + 1)$, where L is the number of leapfrog steps and d is the parameter dimensionality. The term $d + 1$ is the sum of one PDE evaluation to find the log likelihood, and d PDE evaluations to find the numerical derivatives by a first-order differencing scheme with respect to each of the d parameters. For the fluid-dynamics model, $d = 4$ and the mean and standard deviation of L are 23 (13). These statistics are based on an optimum L given by Bayesian optimisation. HMC is run with the number of leapfrog steps drawn from a uniform distribution with lower bound being 1 and upper bound being optimum L.[65] These results are based on measured (physiological) data.

7.4.6 | Summarising conclusions

In conclusion, given that ALDMC registers one of the two highest min(ESS) normalised by the total number of MCMC samples, one of the two highest min(ESS) normalised by the number of forward evaluations, and by far the highest min(ESS) normalised by CPU time, the empirical finding of our study is that in the context of the biophysical model investigated, the ALDMC algorithm with emulation is the best candidate for automatic clinical decision support. This conclusion holds for both physiological and synthetic data. In addition, considering the equivalent number of PDE evaluations replaced by the GP surrogate, it is clear that the emulation approach proposed is a key enabler for using the PDE model as a model-based disease diagnostic in the clinic.

7.5 | Limitations and future work

In the current study, the assumption of a constant stiffness parameter that is independent of the blood vessel dimension (Equation (2)), could be an oversimplification of the true physiological system. Thus, future work will include exploring a more complex physiological model which allows each pulmonary blood vessel to have its own wall stiffness parameter in a Bayesian hierarchical model, with a focus on formal model comparison to select the best model supported by the data. This modification will significantly increase the model complexity, possibly introducing multi-modality in the posterior distribution.

HMC-type algorithms work well for high-dimensional posterior distributions, possibly with high correlations (see Chap. 5 in Reference 23 for a 100-dimensional multivariate Gaussian distribution as a target density). For sampling from multi-modal posteriors, HMC can be coupled with a tempering scheme (e.g., parallel tempering[93]), which entails sampling from a series of tempered distributions that are more diffuse than the target density (with flattened out modes), enabling the sampler to traverse the regions of low probability between the modes.[94]

The increase in parameter complexity could lead to a much larger number of training points required to train the GP emulator to ensure a dense enough coverage of the parameter space. O'Hagan[12] notes that GP emulation is likely to be implemented effectively with up to 50 inputs on modern computing platforms. The number of training points required to optimally cover a 50D input space depends on the complexity of the problem (e.g., smoothness of the log posterior), and is restricted by the $O(n^3)$ computational complexity of GP emulators (due to the covariance matrix inversion). Conventional GPs may be replaced by sparse GPs,[95] which optimally select a lower number of training points while retaining the maximum information at reduced computational costs.

In addition, in future studies we will use GPs to model the correlation of the measurement errors,[6] caused by the temporal nature of the data, and by the smoothing and averaging. We will also model the apparent model mismatch, related to the discrepancy between the real-world system and the simulator output (see the peak shift in the top panel of Figure 5). The model discrepancy may be caused by errors from the numerical integration of the PDEs, the assumption of purely elastic vessel walls, pressure and flow waveforms that do not have a direct physical relationship (due to the data smoothing and averaging), or by the strict periodicity assumption of the measured pressure and flow (Equation (3)).

Ignoring model discrepancy can potentially lead to an underestimation of the uncertainty and a bias in the predictions – see[11,14,96-98] for further details. With the focus on computational efficiency and comparative efficiency evaluation, studying this in more detail is beyond the remit of the present work. However, in future work, we will systematically identify all sources of potential model limitations, including assumptions made about the pulmonary arterial network geometric parameters (e.g., vessels' radii, vessels' lengths), the vessels' connectivity (i.e., vessel bifurcations, trifurcations), and the network size (i.e., number of vessels).[99] We will then follow[14,96] and apply a separate GP to model and quantify this discrepancy. This will lead to a certain increase in the computational complexity, and will thus benefit substantially from the efficiency evaluation and improvement carried out in the present work.

7.6 | Final remarks

In conclusion, biophysical parameters of mathematical cardiovascular models have genuine predictive value for disease prognostication, as demonstrated in Reference 100. However, fast parameter estimation and computational efficiency are paramount for clinical decision making. Our benchmark study comparing several state-of-the-art sampling methods, particularly adapted to complex and expensive computational models sheds light on their relative computational efficiencies. This is a stepping stone towards a decision support system for personalised medicine that can help clinical practitioners to make informed decisions in real time.

Parts of the work reported in this paper were presented by the first author at the annual STEM for BRITAIN competition at the Houses of Parliament in Westminster, UK on March 9, 2020, where they won the gold medal in the category “Mathematics”; see http://www.setforbritain.org.uk/2020event.asp for details.

ACKNOWLEDGEMENTS

This work is part of the research programme of the Centre for Multiscale Soft Tissue Mechanics with Application to Heart & Cancer (SofTMech), funded by the Engineering and Physical Sciences Research Council (EPSRC) of the UK, grant reference number EP/N014642/1. Dirk Husmeier is supported by a grant from the Royal Society of Edinburgh, award number 62335. The authors would like to thank and express their appreciation to Prof. Mette Olufsen, Prof. Nicholas Hill, Dr. Umar Qureshi and Mitchel Colebank for providing the mathematical model and for the fruitful discussions that have led to writing this paper. The data were made available by N. Chesler, Department of Biomedical Engineering, University of Wisconsin, Madison.

ORCID

L. Mihaela Paun https://orcid.org/0000-0002-8734-8135
Dirk Husmeier https://orcid.org/0000-0003-1673-7413

ENDNOTES

[*] First-order finite difference scheme: $\frac{df(x)}{dx} = \frac{f(x+h)-f(x)}{h}$, where $h > 0$ is a small constant.

[†] The PDE system is augmented to include the gradient of the states with respect to the parameters (called the sensitivity derivative equation).

[‡] Avoids solving the sensitivity derivative equation by using a linear adjoint equation, which is easier to numerically integrate.

[§] The likelihood is tractable, thus Bayesian methods such as Approximate Bayesian Computation (ABC)[59] or pseudo-marginal MCMC[60] are not required here.

[¶] In the rare case that a move into the invalid regime gets accepted, a value based on GP extrapolation will be used for the surrogate log posterior. This value is unlikely to be in good agreement with the true log posterior. However, recall from the previous discussion that such a mismatch only affects the computational efficiency of the sampling algorithm, not its accuracy. A poor match between the true and emulated posteriors leads to a low acceptance probability in Equation (14), and thus low efficiency.

[**] For that reason, we replace the squared exponential kernel, which is infinitely differentiable, by the Matérn class kernel, which gives a prior distribution that assumes the functions are only once differentiable.

[††] We checked the analytical derivative against the numerical derivative; a difference in Euclidean space below a small threshold, for example, 0.1%, is taken as an indication that the two are in agreement.

[‡‡] The Bayesian optimisation converges to $\epsilon = 0.005 \pm 0.0004$, $L = 48 \pm 3$ for AHMC, and $\epsilon = 0.078 \pm 0.004$, $L = 38 \pm 4$ for ALDMC.

REFERENCES

1. Huberts W, Heinen SG, Zonnebeld N, et al. What is needed to make cardiovascular models suitable for clinical decision support? A viewpoint paper. *J Comput Sci.* 2018;24:68-84.

2. Hose DR, Lawford PV, Huberts W, Hellevik LR, Omholt SW, van de Vosse FN. Cardiovascular models for personalised medicine: where now and where next? *Med Eng Phys*. 2019;72:38-48.
3. Valdez-Jasso D, Bia D, Zcalo Y, Armentano R, Haider M, Olufsen M. Linear and nonlinear viscoelastic modeling of aorta and carotid pressure-area dynamics under in vivo and ex vivo conditions. *Ann Biomed Eng*. 2011;39:1438-1456.
4. Williams N, Wind-Willassen O, Wright A, et al. Patient specific modeling of headup tilt. *Math Med Biol*. 2014;31:365-392.
5. Lungu A, Wild J, Capener D, Kiely D, Swift A, Hose D. MRI model-based non-invasive differential diagnosis in pulmonary hypertension. *J Biomech*. 2014;47(12):2941-2947.
6. Qureshi M, Colebank M, Mihaela Paun L, et al. A computational study of pulmonary hemodynamics in healthy and hypoxic mice. *Biomech Model Mechanobiol*. 2018;18(1):219-243.
7. Mihaela Paun L, Qureshi M, Colebank M, et al. MCMC methods for inference in a mathematical model of pulmonary circulation. *Statistica Neerlandica*. 2018;72(3):306-338.
8. Tran J, Schiavazzi D, Ramachandra A, Kahnb A, Marsden A. Automated tuning for parameter identification and uncertainty quantification in multi-scale coronary simulations. *Comput Fluids*. 2017;142:128-138.
9. Schiavazzi D, Baretta A, Pennati G, Hsia T, Marsden A. Patient-specific parameter estimation in single-ventricle lumped circulation models under uncertainty. *Int J Numer Method Biomed Eng*. 2017;33(3):e02799.
10. Haario H, Laine M, Mira A, Saksman E. DRAM: efficient adaptive MCMC. *Stat Comput*. 2006;16(4):339-354.
11. Kennedy MC, O'Hagan A. Bayesian calibration of computer models. *J R Stat Soc Series B Stat Methodology*. 2001;63(3):425-464.
12. O'Hagan A. Bayesian analysis of computer code outputs: a tutorial, reliability engineering & system safety. 2006;91(10):1290-1300.
13. Sacks J, Welch W, Mitchell T, Wynn H. Design and analysis of computer experiments. *Stat Sci*. 1989;4(4):409-423.
14. Kennedy M, O'Hagan A. Predicting the output from a complex computer code when fast approximations are available. *Biometrika*. 1998;87:1-13.
15. Mirams GR, Pathmanathan P, Gray RA, Challenor P, Clayton RH. Uncertainty and variability in computational and mathematical models of cardiac physiology. *J Physiol*. 2016;594(23):6833-6847.
16. Wilkinson DJ. Bayesian methods in bioinformatics and computational systems biology. *Brief Bioinform*. 2007;8(2):109-116.
17. Oden JT. Adaptive multiscale predictive modelling. *Acta Numer*. 2018;27:353-450.
18. Hawkins A, Prudhomme S, van der Zee K, Oden J. Bayesian calibration, validation, and uncertainty quantification of diffuse interface models of tumor growth. *J Math Biol*. 2013;67:1457-1485.
19. Akintunde AR, Miller KS, Schiavazzi DE. Bayesian inference of constitutive model parameters from uncertain uniaxial experiments on murine tendons. *J Mech Behav Biomed Mater*. 2019;96:285-300.
20. Harrod KK, Rogers JL, Feinstein JA, Marsden AL, Schiavazzi DE, Predictive modeling of secondary pulmonary hypertension in left ventricular diastolic dysfunction, medRxiv. https://www.medrxiv.org/content/medrxiv/early/2020/04/24/2020.04.23.20073601.full.pdf. Accessed December 16, 2020.
21. Metropolis N, Rosenbluth A, Rosenbluth M, Teller A, Teller E. Equation of state calculations by fast computing machines. *J Chem Phys*. 1953;21(6):1087-1092.
22. Neal R. MCMC using Hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*. Vol 2. New York: Chapman & Hall/CRC Press; 2011:113-162.
23. Brooks S, Gelman A, Jones GL, Meng X-L, eds. *Handbook of Markov Chain Monte Carlo*. New York: Handbooks of Modern Statistical Methods. Chapman & Hall/CRC Press; 2011.
24. Sengupta B, Friston K, Penny W. Gradient-based MCMC samplers for dynamic causal modelling. *NeuroImage*. 2016;125:1107-1118.
25. Kramer A, Calderhead B, Radde N. Hamiltonian Monte Carlo methods for efficient parameter estimation in steady state dynamical systems. *BMC Bioinformatics*. 2014;15(1):253.
26. Lê M, Delingette H, Kalpathy-Cramer J, et al. MRI based Bayesian personalization of a tumor growth model. *IEEE Trans Med Imaging*. 2016;35(10):2329-2339.
27. Bui-Thanh T, Girolami M. Solving large-scale PDE-constrained Bayesian inverse problems with Riemann manifold Hamiltonian Monte Carlo. *Inverse Probl*. 2014;30(11):114014.
28. Vatsa V. Computation of sensitivity derivatives of Navier–Stokes equations using complex variables. *Adv Eng Softw*. 2000;31(8):655-659.
29. Sengupta B, Friston K, Penny W. Efficient gradient computation for dynamical models. *NeuroImage*. 2014;98:521-527.
30. Chen T, Fox E, Guestrin C. (2014). Stochastic Gradient Hamiltonian Monte Carlo. Proceedings of the 31st International Conference on Machine Learning, in PMLR 32(2):1683-1691.
31. Betancourt M. The fundamental incompatibility of scalable Hamiltonian Monte Carlo and Naive data subsampling. In: Bach F, Blei D, eds. *Proceedings of the 32nd International Conference on Machine Learning, Vol. 37 of Proceedings of Machine Learning Research*. Lille, France: PMLR; 2015:533-540.
32. Rasmussen C. Gaussian processes to speed up hybrid Monte Carlo for expensive Bayesian integrals. *Bayesian Stat*. 2003;7(7):651-659.
33. Zhang C, Shahbaba B, Zhao H. Hamiltonian Monte Carlo acceleration using surrogate functions with random bases. *Stat Comput*. 2017;27(6):1473-1490.
34. Cheng Z. Scalable Hamiltonian Monte Carlo via Surrogate Methods, Ph.D. thesis, University of California, Irvine (2016). https://escholarship.org/uc/item/61j792r1. Accessed December 16, 2020.
35. Peirlinck M, Sahli Costabal F, Sack KL, et al. Using machine learning to characterize heart failure across the scales. *Biomech Model Mechanobiol*. 2019;18(6):1987-2001.

36. Schiavazzi D, Arbia G, Baker C, et al. Uncertainty quantification in virtual surgery hemodynamics predictions for single ventricle palliation. *Int J Numer Methods Biomed Eng*. 2016;32(3):e02737.
37. Konukoglu E, Relan J, Cilingir U, et al. Efficient probabilistic model personalization integrating uncertainty on data and parameters: application to eikonal-diffusion models in cardiac electrophysiology. *Prog Biophys Mol Biol*. 2011;107:134-146.
38. Melis A, Clayton R, Marzo A. Bayesian sensitivity analysis of a 1D vascular model with Gaussian process emulators. *Int J Numer Methods Biomed Eng*. 2017;33(12):e2882.
39. Mihaela Paun L, Colebank M, Umar Qureshi M, Olufsen M, Hill N, Husmeier D. MCMC with Delayed Acceptance using a Surrogate Model with an Application to Cardiovascular Fluid Dynamics. Paper presented at: Proceedings of the International Conference on Statistics: Theory and Applications (ICSTA'19), 2019. https://avestia.com/ICSTA2019_Proceedings/files/paper/ICSTA_28.pdf. Accessed December 16, 2020.
40. Quaglino A, Pezzuto S, Koutsourelakis P, Auricchio A, Krause R. Fast uncertainty quantification of activation sequences in patient-specific cardiac electrophysiology meeting clinical time constraints. *Int J Numer Methods Biomed Eng*. 2018;34(7):e2985.
41. Rynn JAJ, Cotter SL, Powell CE, Wright L. Surrogate accelerated Bayesian inversion for the determination of the thermal diffusivity of a material. *Metrologia*. 2019;56(1):015018.
42. Costabal FS, Matsuno K, Yao J, Perdikaris P, Kuhl E. Machine learning in drug development: characterizing the effect of 30 drugs on the QT interval using Gaussian process regression, sensitivity analysis, and uncertainty quantification. *Comput Methods Appl Mech Eng*. 2019;348:313-333.
43. Renardy M, Yi T-M, Xiu D, Chou C-S. Parameter uncertainty quantification using surrogate models applied to a spatial model of yeast mating polarization. *PLoS Comput Biol*. 2018;14(5):e1006181.
44. Bliznyuk N, Ruppert D, Shoemaker C, Regis R, Wild S, Mugunthan P. Bayesian calibration and uncertainty analysis for computationally expensive models using optimization and radial basis function approximation. *J Comput Graph Stat*. 2008;17:270-294.
45. Dietzel A, Reichert P. Bayesian inference of a lake water quality model by emulating its posterior density. *Water Resour Res*. 2014;50 (10):7626-7647.
46. Fielding M, Nott D, Liong S-Y. Efficient MCMC schemes for computationally expensive posterior distributions. *Dent Tech*. 2011;53: 16-28.
47. Wilkinson R. Accelerating ABC methods using Gaussian processes. In: Kaski S, Corander J, eds. *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics, Vol. 33 of Proceedings of Machine Learning Research*. Reykjavik, Iceland: PMLR; 2014:1015-1023.
48. Cotter SL, Dashti M, Stuart AM. Approximation of Bayesian inverse problems for PDEs. *SIAM J Numer Anal*. 2010;48(1):322-345.
49. Wu K, Li J. A surrogate accelerated multicanonical Monte Carlo method for uncertainty quantification. *J Comput Phys*. 2016;321:1098-1109.
50. Conrad PR, Marzouk YM, Pillai NS, Smith A. Accelerating asymptotically exact MCMC for computationally intensive models via local approximations. *J Am Stat Assoc*. 2016;111(516):1591-1607.
51. Conrad PR, Davis AD, Marzouk YM, Pillai NS, Smith A. Parallel local approximation MCMC for expensive models. *SIAM/ASA J Uncert Quantif*. 2018;6(1):339-373.
52. Gong W, Duan Q. An adaptive surrogate modeling-based sampling strategy for parameter optimization and distribution estimation (ASMO-PODE). *Environ Model Softw*. 2017;95:61-75.
53. Christen J, Fox C. Markov chain Monte Carlo using an approximation. *J Comput Graph Stat*. 2005;14(4):795-810.
54. Golightly A, Henderson DA, Sherlock C. Delayed acceptance particle MCMC for exact inference in stochastic kinetic models. *Stat Comput*. 2015;25(5):1039-1055.
55. Sherlock C, Golightly A, Henderson D. Adaptive, delayed-acceptance MCMC for targets with expensive likelihoods. *J Comput Graph Stat*. 2017;26(2):434-444.
56. Higdon D, Reese C, Moulton J, Vrugt J, Fox C. Posterior exploration for computationally intensive forward models. *Handbook of Markov Chain Monte Carlo*. New York: Chapman & Hall/CRC Press; 2011:401-418.
57. Cui T, Fox C, O'Sullivan M. Bayesian calibration of a large-scale geothermal reservoir model by a new adaptive delayed acceptance Metropolis Hastings algorithm. *Water Resour Res*. 2011;47(10):W10521.
58. Lan S, Bui-Thanh T, Christie M, Girolami M. Emulation of higher-order tensors in manifold Monte Carlo methods for Bayesian inverse problems. *J Comput Phys*. 2016;308:81-101.
59. Sunnåker M, Busetto AG, Numminen E, Corander J, Foll M, Dessimoz C. Approximate Bayesian computation. *PLoS Comput Biol*. 2013;9:e1002803.
60. Andrieu C, Roberts GO. The pseudo-marginal approach for efficient Monte Carlo computations. *Ann Stat*. 2009;37(2):697-725.
61. Dhamala J, Arevalo HJ, Sapp J, et al. Quantifying the uncertainty in model parameters using Gaussian process-based Markov chain Monte Carlo in cardiac electrophysiology. *Med Image Anal*. 2018;48:43-57.
62. Hoffman M, Gelman A. The no-U-turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo. *J Mach Learn Res*. 2014; 15(1):1593-1623.
63. Girolami M, Calderhead B. Riemann manifold Langevin and Hamiltonian Monte Carlo methods. *J R Stat Soc Series B Stat Methodology*. 2011;73(2):123-214.
64. Lan S, Stathopoulos V, Shahbaba B, Girolami M. Markov chain Monte Carlo from Lagrangian dynamics. *J Comput Graph Stat*. 2015;24 (2):357-378.

65. Ziyu W, Shakir M, Nando F. Adaptive Hamiltonian and Riemann Manifold Monte Carlo Samplers. Proceedings of the 30th International Conference on Machine Learning, PMLR 2013;28(3):1462-1470.
66. Qureshi M, Haider M, Chesler N, Olufsen M. Simulating effects of hypoxia on pulmonary haemodynamics in mice. Paper presented at: Proc. 5th International Conference on Computational & Mathematical Biomedical Engineering (CMBE 2017), Vol. 1, Zeta Computational Resources Ltd. 2017, pp. 271-274.
67. Yang W, Dong M, Rabinovitch M, Chan FP, Marsden AL, Feinstein JA. Evolution of hemodynamic forces in the pulmonary tree with progressively worsening pulmonary arterial hypertension in pediatric patients. *Biomech Model Mechanobiol*. 2019;18(3):779-796.
68. Bordones AD, Leroux M, Kheyfets VO, Wu YA, Chen CY, Finol EA. Computational fluid dynamics modeling of the human pulmonary arteries with experimental validation. *Ann Biomed Eng*. 2018;46(9):1309-1324.
69. Krenz G, Dawson A. Flow and pressure distributions in vascular networks consisting of distensible vessels. *Am J Phys Heart Circ Phys*. 2003;284(6):H2192–H2203.
70. Tabima D, Roldan-Alzate A, Wang Z, Hacker T, Molther R, Chesler N. Persistent vascular collagen accumulation alters hemodynamic recovery from chronic hypoxia. *J Biomech*. 2012;45(5):799-804.
71. Vanderpool R, Kim A, Molthen R, Chesler N. Effects of acute rho kinase inhibition on chronic hypoxia-induced changes in proximal and distal pulmonary arterial structure and function. *J Appl Physiol*. 2011;110(1):188-198.
72. Conti S, Gosling JP, Oakley JE, O'Hagan A. Gaussian process emulation of dynamic computer codes. *Biometrika*. 2009;96(3):663-676.
73. Conti S, O'Hagan A. Bayesian emulation of complex multi-output and dynamic computer models. *J Stat Plan Inference*. 2010;140(3): 640-651.
74. Rasmussen C, Williams C. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. Cambridge, Massachusetts: The MIT Press; 2005.
75. Mockus J, Tiesis V, Zilinskas A. The application of Bayesian methods for seeking the extremum. *Towards Global Optimization*. Vol 2. Amsterdam; New York: North-Holand; 1978:117-129.
76. Shahriari B, Swersky K, Wang Z, Adams RP, de Freitas N. Taking the human out of the loop: a review of Bayesian optimization. *Proc IEEE*. 2016;104:148-175.
77. Gelbart M, Snoek J, Adams R, Bayesian optimization with unknown constraints. Paper presented at: Proceedings of the Thirtieth Annual Conference on Uncertainty in Artificial Intelligence (UAI-14), AUAI Press, Corvallis, Oregon, 2014, pp. 250–259.
78. Noè U, Chen W, Filippone M, Hill N, Husmeier D. Inference in a partial differential equations model of pulmonary arterial and venous blood circulation using statistical emulation. *Computational Intelligence Methods for Bioinformatics and Biostatistics*. Cham: Springer International Publishing; 2017:184-198.
79. McKay MD, Beckman RJ, Conover WJ. Comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Dent Tech*. 1979;21(2):239-245.
80. Bratley P, Fox B. Algorithm 659: implementing Sobol's Quasirandom sequence generator. *ACM Trans Math Softw*. 1988;14(1):88-100.
81. Bastos LS, O'Hagan A. Diagnostics for Gaussian process emulators. *Dent Tech*. 2009;51(4):425-438.
82. Vanhatalo J, Riihimäki J, Hartikainen J, Jylänki P, Tolvanen V, Vehtari A. GPstuff: Bayesian modeling with Gaussian processes. *J Mach Learn Res*. 2013;14(1):1175-1179.
83. Lax P, Wendroff B. Systems of conservation laws. *Commun Pure Appl Math*. 1960;13(2):217-237.
84. Brooks S, Gelman A. General methods for monitoring convergence of iterative simulations. *J Comput Graph Stat*. 1998;7(4):434-455.
85. Kass R, Carlin B, Gelman A, Neal R. Markov chain Monte Carlo in practice: a roundtable discussion. *Am Stat*. 1998;52(2):93-100.
86. Geyer CJ. Introduction to Markov chain Monte Carlo. *Handbook of Markov Chain Monte Carlo*. New York: Chapman & Hall/CRC; 2011.
87. Cowles MK, Carlin BP. Markov chain Monte Carlo convergence diagnostics: a comparative review. *J Am Stat Assoc*. 1996;91(434): 883-904.
88. Gelman A, Rubin DB. Inference from iterative simulation using multiple sequences. *Stat Sci*. 1992;7(4):457-472.
89. Haario H, Saksman E, Tamminen J. An adaptive Metropolis algorithm. *Ther Ber*. 2001;7(2):223-242.
90. Bowman AW, Azzalini A. *Applied Smoothing Techniques for Data Analysis*. New York, NY: Oxford University Press Inc.; 1997.
91. Dondelinger F, Husmeier D, Rogers S, Filippone M. ODE parameter inference using adaptive gradient matching with Gaussian processes. In: Carvalho CM, Ravikumar P, eds. *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics, Vol. 31 of Proceedings of Machine Learning Research*. Scottsdale, AZ: PMLR; 2013:216-228.
92. Macdonald B, Higham C, Husmeier D. Controversy in mechanistic modelling with Gaussian processes. In: Bach F, Blei D, eds. *Proceedings of the 32nd International Conference on Machine Learning, Vol. 37 of Proceedings of Machine Learning Research*. Lille, France: PMLR; 2015:1539-1547.
93. Geyer CJ, Markov chain Monte Carlo maximum likelihood. Computing Science and Statistics. Paper presented at: Proc. 23rd Symp. on the Interface, Fairfax, 1991, pp. 156–163.
94. Nemeth C, Lindsten F, Filippone M, Hensman J. Pseudo-extended Markov chain Monte Carlo. In: Wallach H, Larochelle H, Beygelzimer A, Alché-Buc F, Fox E, Garnett R, eds. *Advances in Neural Information Processing Systems*. 32. New York: Curran Associates Inc.; 2019:4312-4322.
95. Snelson E, Ghahramani Z. Sparse Gaussian processes using pseudo-inputs. In: Weiss Y, Schölkopf B, Platt JC, eds. *Advances in Neural Information Processing Systems 18*. Cambridge, MA: MIT Press; 2006:1257-1264.

96. Brynjarsdóttir J, O'Hagan A. Learning about physical parameters: the importance of model discrepancy. *Inverse Probl*. 2014;30(11): 114007.
97. Wang J-X, Roy CJ, Xiao H. Propagation of input uncertainty in presence of model-form uncertainty: a multifidelity approach for computational fluid dynamics applications. *ASCE-ASME J Risk Uncert Eng Sys Part B Mech Eng*. 2018;4(1):011002.
98. Lei CL, Ghosh S, Whittaker DG, et al. Considering discrepancy when calibrating a mechanistic electrophysiology model, philosophical transactions of the Royal Society a: mathematical. *Phys Eng Sci*. 2020;378(2173):20190349.
99. Colebank MJ, Paun LM, Qureshi MU, et al. Influence of image segmentation on one-dimensional fluid dynamics predictions in the mouse pulmonary arteries. *J R Soc Interface*. 2019;16(159):20190284.
100. Gao H, Aderhold A, Mangion K, Luo X, Husmeier D, Berry C. Changes and classification in myocardial contractile function in the left ventricle following acute myocardial infarction. *J R Soc Interface*. 2017;14(132):20170203.
101. Schaul T. *Optimization with Surrogate Models*. Dordrecht: Springer Netherlands; 2013:55-70.
102. Shahsavani D, Tarantola S, Ratto M. Evaluation of MARS modeling technique for sensitivity analysis of model output. *Procedia Soc Behav Sci*. 2010;2(6):7737-7738. Sixth International Conference on Sensitivity Analysis of Model Output.
103. Chandra R, Azam D, Kapoor A, Mulller R. Surrogate-assisted Bayesian inversion for landscape and basin evolution models, Geoscientific Model Development Discussions (2019) 1–28.
104. Pruett W, Hester R. The creation of surrogate models for fast estimation of complex model outcomes. *PLoS One*. 2016;11:e0156574.
105. Razavi S, Tolson B, Burn D. Review of surrogate modeling in water resources. *Water Resour Res*. 2012;48(7):W07401.
106. Asher MJ, Croke BFW, Jakeman AJ, Peeters LJM. A review of surrogate models and their application to groundwater modeling. *Water Resour Res*. 2015;51(8):5957-5973.
107. Alden K, Cosgrove J, Coles M, Timmis J. Using emulation to engineer and understand simulations of biological systems. *IEEE/ACM Trans Comput Biol Bioinform*. 2020;17(1):302-315.
108. Taflanidis AA, Zhang J, Patsialis D. Applications of reduced order and surrogate modeling in structural dynamics. In: Barthorpe R, ed. *Model Validation and Uncertainty Quantification*. Vol 3. Switzerland: Springer International Publishing; 2020:297-299.
109. Frangos M, Marzouk Y, Willcox K, van Bloemen Waanders B. Surrogate and Reduced-Order Modeling: A Comparison of Approaches for Large-Scale Statistical Inverse Problems. *Large-Scale Inverse Problems and Quantification of Uncertainty*. England: John Wiley & Sons, Ltd; 2010, Chap. 7:123-149.
110. Quaglino A, Pezzuto S, Krause R. High-dimensional and higher-order multifidelity Monte Carlo estimators. *J Comput Phys*. 2019;388: 300-315.
111. Ebden M. Gaussian Processes: A Quick Introduction, ArXiv e-prints, arXiv:1505.02965.
112. Holmes CC, Held L. Bayesian auxiliary variable models for binary and multinomial regression. *Bayesian Anal*. 2006;1(1):145-168.
113. Williams CKI, Barber D. Bayesian classification with Gaussian processes. *IEEE Trans Pattern Anal Mach Intell*. 1998;20(12):1342-1351.
114. Nguyen TV, Bonilla EV. Automated variational inference for Gaussian process models. In: Ghahramani Z, Welling M, Cortes C, Lawrence ND, Weinberger KQ, eds. *Advances in Neural Information Processing Systems 27*. New York: Curran Associates, Inc; 2014: 1404-1412.
115. Kuss M. Gaussian Process Models for Robust Regression, Classification, and Reinforcement Learning, Ph.D. thesis, Technische Universität Darmstadt (2006).
116. Chaari L, Tourneret J, Chaux C, Batatia H. A Hamiltonian Monte Carlo method for non-smooth energy sampling. *IEEE Trans Signal Process*. 2016;64(21):5585-5594.
117. McClarren RG. *Uncertainty Quantification and Predictive Computational Science: A Foundation for Physical Scientists and Engineers*. Switzerland: Springer International Publishing; 2018.

How to cite this article: Paun LM, Husmeier D. Markov chain Monte Carlo with Gaussian processes for fast parameter estimation and uncertainty quantification in a 1D fluid-dynamics model of the pulmonary circulation. *Int J Numer Meth Biomed Engng*. 2021;37:e3421. https://doi.org/10.1002/cnm.3421

APPENDIX

Brief review of surrogate modelling

Surrogate models can be split into three categories:

- Data-driven models (data fit interpolation and regression models) are constructed by an input–output map for the original (high-fidelity) model, and methods to do this include polynomial chaos expansion,[41] Gaussian Processes (GPs), also known as Kriging,[35,42,72-74,101] multivariate adaptive regression splines,[102] random basis functions,[33]neural networks,[103] support vector machines[104]; for a review on data-driven surrogate models, the reader is referred to.[105-107]

- Projection-based reduced models reduce the parameter space dimensionality by projecting the governing equations into a basis of orthonormal vectors – see[106] for a review and[108,109] for an application to structural dynamics, and to large-scale statistical inverse problems.
- Multi-fidelity (hierarchical) models are surrogate models created from the original (high-fidelity, complex) model by reducing the numerical resolution, or by simplifying the physical process – see[106] for a review and[40,110] for an application to cardiac electrophysiology.

Given that the fluid-dynamics model under consideration has a fairly low number of parameters (4), in our work we use GPs, a probabilistic inference method reviewed in Appendix A.2, to construct a surrogate for the data log likelihood. We note that the data likelihood points used to construct the surrogate model are calculated based on the original, high-fidelity fluid-dynamics model (i.e., the original governing fluid-dynamics equations are used).

Gaussian processes

This section gives a brief overview of GPs. For a detailed introduction to GPs, the reader is referred to.[74]

Gaussian processes for regression

A stochastic process $f = f(\mathbf{x})_{\mathbf{x}\in\mathcal{X}}$ is defined as a Gaussian process if the random variables $\mathbf{f} = (f(\mathbf{x}_1), ..., f(\mathbf{x}_n))$ are jointly normal for any inputs $\mathbf{x}_i \in \mathbb{R}^{d\times 1}$, with $i = 1, ..., n$, $\mathbf{f} \sim \mathcal{MVN}(\mathbf{m}, \mathbf{K})$, where $\mathbf{m} = (m(\mathbf{x}_1), ..., m(\mathbf{x}_n))$ is the mean n-vector and $\mathbf{K} = \left[k\left(\mathbf{x}_i, \mathbf{x}_j\right)\right]_{i,j=1}^{n}$ is the $n \times n$ variance–covariance matrix of $\mathbf{f}$. In GP models,[74,111] inputs $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_n]^T$ ($\mathbf{X}$ is an $n \times d$ matrix) are mapped into outputs $\mathbf{y} = (y_1, ..., y_n)$ ($\mathbf{y}$ is an n-vector) by means of latent noiseless functions $\mathbf{f}$. A GP prior is placed on the distribution of these functions as a way to account for the uncertainty in the functional form. Assuming iid Gaussian noisy observations $\mathbf{y}$:

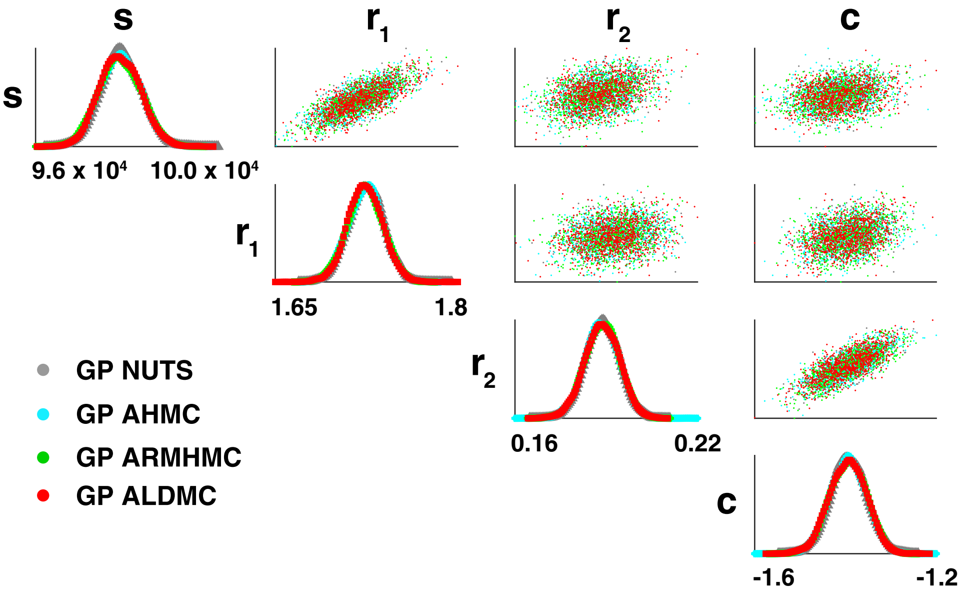

FIGURE A1 Measured (physiological) data inference results. Marginal posterior densities obtained with kernel density estimation, as well as the pairwise scatterplots, based on the posterior samples drawn using all emulation Gaussian Process Hamiltonian and Lagrangian Dynamical Monte Carlo algorithms. Each simulation consisted of 5000 sampling phase draws and 500 burn-in draws (the latter were discarded). Legend: GP: Gaussian Process; NUTS: No U-turn Sampler; AHMC: Adaptive Hamiltonian Monte Carlo; ARMHMC: Adaptive Riemann Manifold Hamiltonian Monte Carlo; ALDMC: Adaptive Lagrangian Dynamical Monte Carlo

$$\begin{aligned}\mathbf{y}\,|\,\mathbf{f} &\sim \mathscr{MVN}\left(\mathbf{f},\sigma^2\mathbf{I}\right),\\ \mathbf{f}(\mathbf{X})\,|\,\boldsymbol{\gamma} &\sim \mathcal{GP}(\mathbf{m}(\mathbf{X}),\mathbf{K}|\boldsymbol{\gamma}),\\ \boldsymbol{\gamma},\sigma^2 &\sim p(\boldsymbol{\gamma})p\left(\sigma^2\right),\end{aligned} \tag{A1}$$

where $\boldsymbol{\gamma}$ contains the covariance function (kernel) hyperparameters, and σ^2 is the observation noise variance. Typically, the mean of the GP is set to zero ($\mathbf{m}(\mathbf{X}) = \mathbf{0}$), that is, the data are standardised to zero mean, which will henceforth be assumed. The covariance function, $k(\mathbf{x}, \mathbf{x}' \mid \boldsymbol{\gamma})$, gives the smoothness and variability of the latent functions. Several covariance functions are available; see for example, Chapter 4 in Reference 74 for a review. The kernel hyperparameters $\boldsymbol{\gamma}$ and the observation noise variance σ^2 can be placed a prior on and integrated out from the joint posterior distribution $p(\mathbf{f}, \boldsymbol{\gamma}, \sigma^2| D)$, where $D = \{\mathbf{X}, \mathbf{y}\}$ is the set of training points, or they can be found in a so-called empirical Bayes approach by maximisation of the log marginal likelihood,

$$\log p\left(\mathbf{y}|\mathbf{X},\boldsymbol{\gamma},\sigma^2\right) = -\frac{n}{2}\log(2\pi) - \frac{1}{2}\log|\mathbf{K}+\sigma^2\mathbf{I}| - \frac{1}{2}\mathbf{y}^T\left(\mathbf{K}+\sigma^2\mathbf{I}\right)^{-1}\mathbf{y}. \tag{A2}$$

The aim of constructing a probabilistic model using GPs is prediction at unseen input values. The predictive distribution of a new function, $\tilde{f} = f(\tilde{\mathbf{x}})$, is also Gaussian,[74] as follows:

$$\tilde{f}\,|\,D,\boldsymbol{\gamma},\sigma^2 \sim \mathcal{N}\left(m_p(\tilde{\mathbf{x}}),k_p\left(\tilde{\mathbf{x}},\tilde{\mathbf{x}}'|\boldsymbol{\gamma}\right)\right), \tag{A3}$$

$$m_p(\tilde{\mathbf{x}}) = \boldsymbol{k}(\tilde{\mathbf{x}},\mathbf{X}|\boldsymbol{\gamma})\left(\mathbf{K}+\sigma^2\mathbf{I}\right)^{-1}\mathbf{y}, \tag{A4}$$

$$k_p\left(\tilde{\mathbf{x}},\tilde{\mathbf{x}}'|\boldsymbol{\gamma}\right) = k\left(\tilde{\mathbf{x}},\tilde{\mathbf{x}}'|\boldsymbol{\gamma}\right) - \boldsymbol{k}(\tilde{\mathbf{x}},\mathbf{X}|\boldsymbol{\gamma})\left(\mathbf{K}+\sigma^2\mathbf{I}\right)^{-1}\boldsymbol{k}\left(\mathbf{X},\tilde{\mathbf{x}}'\,|\,\boldsymbol{\gamma}\right), \tag{A5}$$

where $\boldsymbol{k}(\tilde{\mathbf{x}},\mathbf{X}|\boldsymbol{\gamma})$ is a vector valued kernel function, $\boldsymbol{k}(\tilde{\mathbf{x}},\mathbf{X}) : \mathbb{R}^{d\times 1}\times\mathbb{R}^{d\times n}\to\mathbb{R}_+^{1\times n}$, $\boldsymbol{k}(\mathbf{X},\tilde{\mathbf{x}}\,|\,\boldsymbol{\gamma})$ is a vector valued kernel function, $\boldsymbol{k}(\mathbf{X},\tilde{\mathbf{x}}) : \mathbb{R}^{d\times n}\times\mathbb{R}^{d\times 1}\to\mathbb{R}_+^{n\times 1}$, and the hyperparameters $\boldsymbol{\gamma}$ and σ^2 have been obtained based on the training data ($\mathbf{X}$, $\mathbf{y}$).

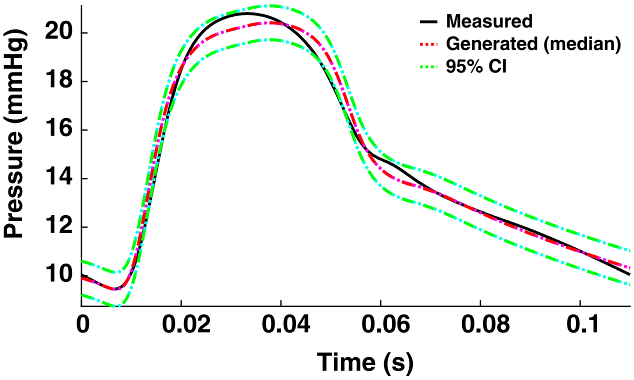

FIGURE A2 Superimposition of posterior median blood pressure time series obtained using all four emulation Hamiltonian and Lagrangian Dynamical Monte Carlo algorithms: GP AHMC, GP NUTS, GP ARMHMC, GP ALDMC (in red/magenta colours and different line types depending on the sampling algorithm used: red dotted/magenta dotted line for NUTS/HMC and red dashed/magenta dashed line for LDMC/RMHMC). With green/cyan colour and different line types depending on the sampling algorithm used (green dotted/cyan dotted line for NUTS/HMC and green dashed/cyan dashed line for LDMC/RMHMC) we show the 95% predictive credible interval, calculated analytically as $m(\boldsymbol{\theta}) \pm 1.96\sigma_{\text{total}}$, where $m(\boldsymbol{\theta})$ is the mean posterior pressure signal, and $\sigma_{\text{total}} = \sqrt{\sigma^2_{\text{error}} + \sigma^2_{\text{signal}}}$, with σ^2_{error} being the mean posterior estimate for the noise variance, and σ^2_{signal} being the variance of the distribution of the signals, which are model predictions obtained with the MCMC posterior parameter samples. Measured (physiological) blood pressure time series (black solid line) is superimposed. Note that the different medians and CIs corresponding to different algorithms are nearly indistinguishable, indicating great similarity in the results produced by the four methods

The derivative of the posterior predictive mean and variance with respect to the new input point $\tilde{\mathbf{x}} = (\tilde{x}_1, ..., \tilde{x}_d)$ can be computed analytically, and for example, the first order partial derivatives are:

$$\frac{\partial m_p(\tilde{\mathbf{x}})}{\partial \tilde{x}_j} = \frac{\partial \boldsymbol{k}(\tilde{\mathbf{x}}, \mathbf{X}|\boldsymbol{\gamma})}{\partial \tilde{x}_j} \left(\mathbf{K} + \sigma^2 \mathbf{I}\right)^{-1} \mathbf{y}, \tag{A6}$$

$$\frac{\partial k_p(\tilde{\mathbf{x}}, \tilde{\mathbf{x}}|\boldsymbol{\gamma})}{\partial \tilde{x}_j} = \frac{\partial k(\tilde{\mathbf{x}}, \tilde{\mathbf{x}}|\boldsymbol{\gamma})}{\partial \tilde{x}_j} - \frac{\partial \boldsymbol{k}(\tilde{\mathbf{x}}, \mathbf{X}|\boldsymbol{\gamma})}{\partial \tilde{x}_j} \left(\mathbf{K} + \sigma^2 \mathbf{I}\right)^{-1} \boldsymbol{k}(\mathbf{X}, \tilde{\mathbf{x}} \mid \boldsymbol{\gamma}) - \boldsymbol{k}(\tilde{\mathbf{x}}, \mathbf{X}|\boldsymbol{\gamma}) \left(\mathbf{K} + \sigma^2 \mathbf{I}\right)^{-1} \frac{\partial \boldsymbol{k}(\mathbf{X}, \tilde{\mathbf{x}} \mid \boldsymbol{\gamma})}{\partial \tilde{x}_j}, \tag{A7}$$

where

$$\left(\frac{\partial \boldsymbol{k}(\tilde{\mathbf{x}}, \mathbf{X}|\boldsymbol{\gamma})}{\partial \tilde{x}_j}\right)^T = \begin{bmatrix} \frac{\partial \boldsymbol{k}(\tilde{\mathbf{x}}, \mathbf{x_1}|\boldsymbol{\gamma})}{\partial \tilde{x}_j} \\ \vdots \\ \frac{\partial \boldsymbol{k}(\tilde{\mathbf{x}}, \mathbf{x_n}|\boldsymbol{\gamma})}{\partial \tilde{x}_j} \end{bmatrix} \quad \text{and} \quad \frac{\partial \boldsymbol{k}(\mathbf{X}, \tilde{\mathbf{x}} \mid \boldsymbol{\gamma})}{\partial \tilde{x}_j} = \begin{bmatrix} \frac{\partial \boldsymbol{k}(\mathbf{x_1}, \tilde{\mathbf{x}}|\boldsymbol{\gamma})}{\partial \tilde{x}_j} \\ \vdots \\ \frac{\partial \boldsymbol{k}(\mathbf{x_n}, \tilde{\mathbf{x}}|\boldsymbol{\gamma})}{\partial \tilde{x}_j} \end{bmatrix}. \tag{A8}$$

Derivatives may be needed if GPs are coupled with methods which require knowledge of the derivatives, for example, the GPHMC algorithm.[32]

In the present paper, we follow the work of[11,72,73] and use GPs for statistical emulation of the unnormalised log posterior distribution, as described in more detail in Section 4.

Gaussian processes for classification

Besides recording a real-valued dependent variable, we may record a binary variable $\boldsymbol{\lambda} = (\lambda_1, ..., \lambda_n)$ with $\lambda_i \in \{0, 1\}$ associated with inputs $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^n$, where 0: failure and 1: success. GPs can also be applied to classification problems, where we are for example interested in deciding on the success of the PDE evaluation for a particular parameter configuration. To do so, we can build a GP classification model that places a GP prior over the distribution of noiseless latent functions, $\mathbf{f}$, as follows:

$$\begin{aligned} &\lambda_i \mid f(\mathbf{x}_i) \sim \text{Bernoulli}(p(\lambda_i = 1 | f(\mathbf{x}_i))), \\ &\mathbf{f}(\mathbf{X}) \mid \boldsymbol{\gamma} \sim \mathcal{GP}(\mathbf{m}(\mathbf{X}), \mathbf{K}|\boldsymbol{\gamma}), \\ &\boldsymbol{\gamma} \sim p(\boldsymbol{\gamma}), \end{aligned} \tag{A9}$$

where $\boldsymbol{\gamma}$ contains the kernel hyperparameters.

The binary observations (class labels), $\boldsymbol{\lambda}$, are drawn from a Bernoulli distribution with a success probability $p(\lambda_i = 1 | f(\mathbf{x}_i))$, and likelihood shown in Equation (A11). The success probability is related to the function $f(\mathbf{x}_i)$ via the sigmoid function, $sig(f(\mathbf{x}_i)) = (1 + \exp(-f(\mathbf{x}_i)))^{-1}$, which transforms the probability into the unit interval, [0,1], as shown in Equation (A10):

$$p(\lambda_i = 1 | f(\mathbf{x}_i)) = \text{sig}(f(\mathbf{x}_i)); \quad p(\lambda_i = 0 | f(\mathbf{x}_i)) = 1 - p(\lambda_i = 1 | f(\mathbf{x}_i)), \tag{A10}$$

$$\text{Likelihood}: p(\lambda_i | f(\mathbf{x}_i)) = (\text{sig}(f(\mathbf{x}_i)))^{\lambda_i} (1 - \text{sig}(f(\mathbf{x}_i)))^{(1-\lambda_i)} \text{ for a binary outcome } \lambda_i \in \{0, 1\}. \tag{A11}$$

The latent functions, $\mathbf{f}$, can be integrated out from the conditional posterior distribution given the hyperparameters $\boldsymbol{\gamma}$, to obtain the marginal likelihood (Equation (A12)), which has no closed-form solution:

$$p(\lambda|\mathbf{X},\boldsymbol{\gamma}) = \int p(\lambda,\mathbf{f}|\mathbf{X},\boldsymbol{\gamma})d\mathbf{f} = \int p(\lambda|\mathbf{f})p(\mathbf{f}|\mathbf{X},\boldsymbol{\gamma})d\mathbf{f}. \tag{A12}$$

With a non-Gaussian likelihood, we can run MCMC,[112] or approximate the conditional posterior distribution by a Gaussian form using the Laplace approximation,[113] variational inference[114] or expectation propagation (EP).[115]

Log posterior differentiability for HMC

In HMC-type algorithms, the log posterior distribution must be differentiable with respect to the parameters. For example, in some cardiovascular models with parameters related to the heart valves and the timing wave propagation phenomena, the log posterior, while being a continuous function, may not be differentiable with respect to the parameters. This issue can be overcome by using an HMC method with a modified leapfrog scheme, in which the gradient of the log posterior is replaced by the proximity operator.[116] Non-differentiability is only an issue if HMC without emulation is employed, whereas when HMC is coupled with emulation, smoothness, thus differentiability, is controlled by the choice of the GP kernel.

For GPHMC, calculating the log posterior derivatives requires finding derivatives of the GP predictive mean and variance, and these expressions are available in analytical form (Equations (A6) and (A7)). In contrast, HMC requires

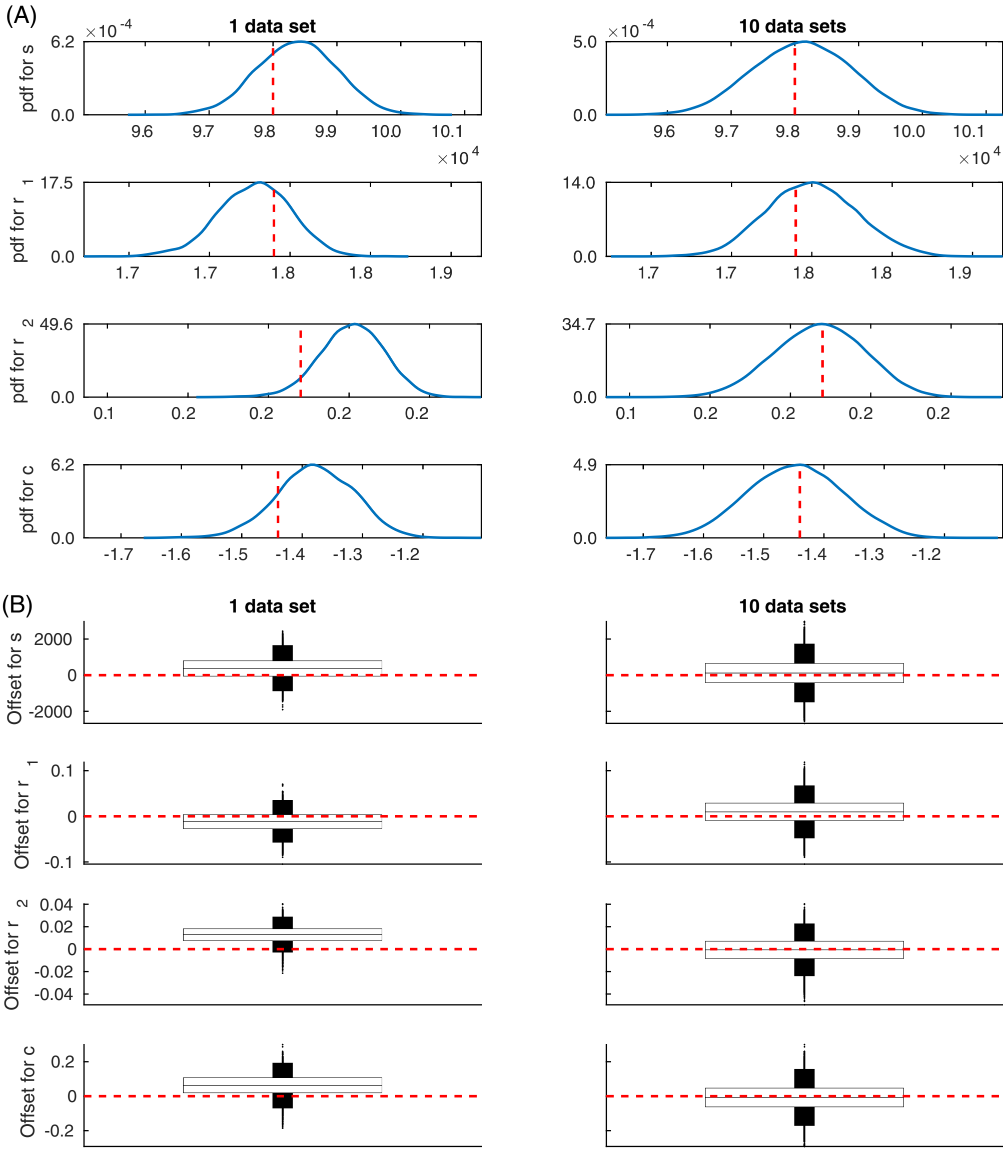

FIGURE A3 Synthetic data inference results. Panel (A): Kernel density estimation plots for the marginal posterior probability densities of the four parameters (s: stiffness, r_1, r_2, c: Windkessel parameters). Five thousand posterior samples are drawn using the GP emulation Hamiltonian Monte Carlo algorithm for every synthetic data set. The left side plots are based on the posterior samples from 1 data set, and the right side plots are based on agglomerated posterior samples from 10 data sets. The true parameter values are superimposed as vertical red lines. Panel (B): Boxplot representation of the distribution of offsets for the parameters, calculated using Equation (A13) for 1 and 10 data sets. The horizontal red line marks zero offset

derivatives for the simulator, which are in most cases obtained numerically (provided differentiability holds), and that requires several considerations to be made (e.g., differencing scheme used: first/second order, choice of discretisation step). These choices may affect the performance of the HMC algorithm, for example for rough signals, the use of first-order differencing (which is less accurate than central differences, i.e., second-order) could lead to a sub-optimal HMC sampler, with reduced exploration efficiency, since unreliable derivative estimates introduce noise into the Hamiltonian equations. For further details, for example, on selecting the discretisation step in the finite difference scheme, see Chap. 4 and fig. 4.5 in Reference 117. These issues only need considering when HMC without emulation is used, however they are avoided in our work, where HMC is coupled with emulation using GPs, since the derivatives are available in analytical form.

Parameter pairwise scatterplots

Figure A1 shows the marginal posterior densities and the pairwise scatterplots, obtained based on the posterior samples drawn using all emulation Gaussian Process Hamiltonian and Lagrangian Dynamical Monte Carlo algorithms for the measured data study.

Model predictions and UQ

Figure A2 displays the model predictions (posterior median signals), as well as the 95% credible intervals obtained with each of the emulation Gaussian Process Hamiltonian and Lagrangian Dynamical Monte Carlo algorithms.

Parameter inference and UQ for multiple synthetic data sets

Figure 7 presents results based on one synthetic data set. To generate the data set, one noise instantiation was added to the prediction from the PDE model. With one noise realisation, we may obtain an offset from the true parameter values. This offset is expected to be increasingly reduced when computed based on agglomerated posterior samples from multiple data sets generated with different noise instantiations. Thus, in expectation (over different data sets), the true parameter values are recovered

To demonstrate this, we have performed the inference analysis by running the emulation HMC algorithm on 10 different data sets generated with different noise instantiations. We combine the posterior samples from all data sets, and the aim is to illustrate that as a larger number of data sets is used, the parameters are more accurately estimated. In Figure A3(A) we display the marginal posterior probability density of the parameters obtained with kernel density estimation. The left side plots are based on posterior samples from 1 data set, and the right side plots are based on agglomerated posterior samples from 10 data sets. Additionally, we calculate the offset for the 4-dimensional parameter vector $\boldsymbol{\theta}$ for 1 and 10 data sets, by subtracting the true parameter from the agglomerated posterior samples (i.e., posterior samples combined from all K data sets) as follows:

$$\mathbf{d}(\boldsymbol{\theta}) = \boldsymbol{\theta}_{\cup_{k=1}^{K}} - \boldsymbol{\theta}_{true}, \quad K = \{1, 10\}, \tag{A13}$$

where $\boldsymbol{\theta}_{\cup_{k=1}^{K}}$ is the union of K sets of parameter posterior samples corresponding to the K data sets. In Figure A3(B) we show the distribution of offsets.

A clear offset is registered when one single data set is used, and this offset gets reduced when the agglomerated posterior samples from 10 data sets are shown. It is straightforward to prove for analytically tractable toy problems (e.g., a binomial distribution used to describe tossing a coin) that the offset converges to zero as the number of independent data instantiations goes to infinity. In principle this could be shown numerically for our problem by increasing the number of independent data instantiations, but the computational costs would be excessive.