ROYAL STATISTICAL SOCIETY
DATA | EVIDENCE | DECISIONS

Journal of the Royal Statistical Society
Applied Statistics
Series C

*Appl. Statist.* (2019)
**68**, *Part* 5, *pp.* 1555–1576

# Fast parameter inference in a biomechanical model of the left ventricle by using statistical emulation

Vinny Davies,

*University of Glasgow, UK*

Umberto Noè,

*German Centre for Neurodegenerative Diseases, Bonn, Germany*

Alan Lazarus, Hao Gao and Benn Macdonald,

*University of Glasgow, UK*

Colin Berry

*University of Glasgow and West of Scotland Heart and Lung Centre, Clydebank, UK*

and Xiaoyu Luo and Dirk Husmeier

*University of Glasgow, UK*

[Received January 2018. Revised July 2019]

**Summary.** A central problem in biomechanical studies of personalized human left ventricular modelling is estimating the material properties and biophysical parameters from *in vivo* clinical measurements in a timeframe that is suitable for use within a clinic. Understanding these properties can provide insight into heart function or dysfunction and help to inform personalized medicine. However, finding a solution to the differential equations which mathematically describe the kinematics and dynamics of the myocardium through numerical integration can be computationally expensive. To circumvent this issue, we use the concept of emulation to infer the myocardial properties of a healthy volunteer in a viable clinical timeframe by using *in vivo* magnetic resonance image data. Emulation methods avoid computationally expensive simulations from the left ventricular model by replacing the biomechanical model, which is defined in terms of explicit partial differential equations, with a surrogate model inferred from simulations generated before the arrival of a patient, vastly improving computational efficiency at the clinic. We compare and contrast two emulation strategies: emulation of the computational model outputs and emulation of the loss between the observed patient data and the computational model outputs. These strategies are tested with two interpolation methods, as well as two loss functions. The best combination of methods is found by comparing the accuracy of parameter inference on simulated data for each combination. This combination, using the output emulation method, with local Gaussian process interpolation and the Euclidean loss function, provides accurate parameter inference in both simulated and clinical data, with a reduction in the computational cost of about three orders of magnitude compared with numerical integration of the differential equations by using finite element discretization techniques.

*Keywords*: Emulation; Gaussian processes; Holzapfel–Ogden constitutive law; Left ventricle heart model; Magnetic resonance imaging; Optimization; Simulation

*Address for correspondence*: Dirk Husmeier, School of Mathematics and Statistics, University of Glasgow, University Place, Glasgow, G12 8SQ, UK.
E-mail: Dirk.Husmeier@glasgow.ac.uk

 0035–9254/19/681555
Published by John Wiley & Sons Ltd on behalf of the Royal Statistical Society.

## 1. Introduction

It is widely recognized that, when integrated with *in vivo* data from cardiac magnetic resonance imaging (MRI), computational modelling of cardiac biomechanics can provide unique insights into cardiac function in both healthy and diseased states (Wang *et al.*, 2015; Chabiniok *et al.*, 2016; Gao, Aderhold, Mangion, Luo, Husmeier and Berry, 2017). For example, recent mathematical studies have demonstrated that passive myocardial stiffness is much higher in diastolic heart failure patients compared with healthy subjects (Xi *et al.*, 2014). Similarly, myocardial contractility could be much higher in acute myocardial infarction patients than it is in healthy volunteers (Gao, Aderhold, Mangion, Luo, Husmeier and Berry, 2017). In particular, the myocardial passive properties not only affect left ventricular (LV) diastolic filling but also influence the pumping function in heart chamber contractions (systole) through the 'Frank–Starling' law (Widmaier *et al.*, 2016), the relationship between stroke volume and end diastolic volume.

To assess LV function comprehensively, it is necessary to determine passive myocardial stiffness. Traditionally myocardial passive properties can be determined by a series of *ex vivo* or *in vitro* experiments (Dokos *et al.*, 2002). The widely used Holzapfel–Ogden (HO) constitutive law (Holzapfel and Ogden, 2009) can give a detailed description of the myocardium response in passive state, including the effects of collagen fibre structure. However, determining the material parameters of this model is challenging for clinical applications, as one cannot perform invasive experiments as in Dokos *et al.* (2002). One possibility of estimating these parameters non-invasively is by cardiac MRI, which allows both early and end diastolic states to be measured. We can then compare, for a given patient, these measurements with the predictions from the biomechanical model, which defines the likelihood. The biophysical parameters defining the myocardial properties (as described by the HO law) can then be inferred in an approximate maximum likelihood sense by using an iterative optimization procedure, as discussed in Gao *et al.* (2015). In the context of mathematical physiology, this procedure is referred to as solving the inverse problem.

The inverse problem itself can be solved by using a variety of methods and many studies have demonstrated that it is possible to estimate constitutive material parameters by using *in vivo* measurements even with very complex constitutive relations (Guccione *et al.*, 1991; Remme *et al.*, 2004; Sermesant *et al.*, 2006; Sun *et al.*, 2009). However, because of the strong correlation between the material parameters and sparse noisy data, the formulated inverse problem is highly non-linear (Xi *et al.*, 2011; Gao *et al.*, 2015). Furthermore, determining the unknown parameters in this way is very time consuming, with the process taking days or weeks to converge, even with a modern multicore workstation (Gao *et al.*, 2015; Nikou *et al.*, 2016). The primary reason for this is the high computational expense of simulating from the biomechanical model, which requires a numerical integration of the underlying partial differential equations with finite element discretization. This procedure must be repeated hundreds or thousands of times during the iterative optimization of the material parameters.

As a result of the high computational costs of simulating the biomechanical model, estimating myocardial properties by using a process which uses this model as a simulator is not suitable for realtime clinical diagnosis. A potential approach to overcome this problem is emulation (e.g. Kennedy and O'Hagan (2001), Conti *et al.* (2009) and Conti and O'Hagan (2010)), which has recently been explored in the closely related contexts of cardiovascular fluid dynamics (Melis *et al.*, 2017), the pulmonary circulatory system (Noè *et al.*, 2017) and ventricular mechanics (Achille *et al.*, 2018).

Emulation methods are far more computationally efficient as most of the computation can be done in advance, making the in-clinic diagnosis faster. With emulation approaches, we

simulate a large number of samples at different parameter specifications in advance and use these simulations combined with an interpolation method to replace the computationally expensive simulator in the optimization procedure. The choice of parameter combinations from which simulations are taken can be determined effectively by using a space filling design, in this case produced by a Sobol sequence (Sobol, 1967), to spread the parameter combinations chosen in a way that aims to maximize the information about the simulator for a given number of simulations via several uniformity conditions. Optimizing this design is an active research area (see for example Overstall and Woods (2017)), which is beyond the remit of the present paper though.

The work that is presented here is designed as a proof-of-concept study to assess the accuracy of alternative emulation strategies for learning the material properties of a healthy volunteer's LV myocardium based on only non-invasive, *in vivo* MRI data. For that, we use a patient-specific model with a fixed, patient-specific LV geometry, and focus on the statistical methodology for biophysical parameter estimation. Additionally, we use a reduced parameterization of the HO law with the biomechanical model based on the work of Gao *et al.* (2015) in MRI data. On the basis of this approach, we compare different emulation strategies, loss functions and interpolation methods.

The first of the emulation approaches that we have tested is based on emulating the outputs of the simulator (Section 3.3.1), in this case the simulated clinical data based on the biomechanical model described. Here, individual interpolators are fitted to each of the simulator outputs, using our chosen interpolation technique. We can then calculate the loss function between the predicted output of the individual models and the observed new data points from which we wish to learn the underlying myocardial properties. Minimizing this loss function via a standard optimization routine then produces estimates of the material parameters of the new subject. A variety of loss functions can be used within our emulation methods and we have compared two different functions here. The first of these is the Euclidean loss function, which assumes independence between outputs, and the second is the Mahalanobis loss function (Mahalanobis, 1936) which allows for correlations.

The second emulation approach involves emulating a loss function rather than the outputs directly (Section 3.3.2), where again we use both the Euclidean and the Mahalanobis loss functions. For new MRI data, we calculate the loss, which quantifies the discrepancy between the model predictions and the data. Statistical interpolation is then used to obtain a surrogate loss function over the biophysical parameter space, which can be minimized with standard iterative optimization routines.

In addition to testing these two emulation paradigms, we test two interpolation techniques based on Gaussian processes (GPs) (Rasmussen and Williams, 2006). The first of these is a low rank GP emulation method, which uses the complete data set for interpolation but uses a low rank approximation to scale to high dimensions (Wood, 2003). The second method uses a local GP, where the interpolation is based on the $K$-nearest-neighbours that are closest to the current values of the material parameters. Using a reduced number of training points from the simulations at each stage of the optimization procedure and thereby lowering the computational costs is important, as because of the cubic computational complexity in the number of training points a standard GP would not be suitable for clinical decision support in realtime.

In this work, we firstly compare different combinations of emulation methods, interpolation methods and loss functions to determine which method provides the best estimate of the material LV properties. We do this via a simulation study (Sections 5.1, 5.2, 5.3 and 5.4), using additional independent simulations from the simulator as out-of-sample test data. Knowledge of the true parameter values enables us to assess the accuracy of the different combinations of methods. We

then test the best combination of methods on real MRI data from the healthy volunteer from whom we have taken the LV geometry (Section 5.5), to assess the accuracy of biomechanical constitutive parameter estimation in a timeframe that is suitable for clinical applications.

## 2. Left ventricle biomechanical model

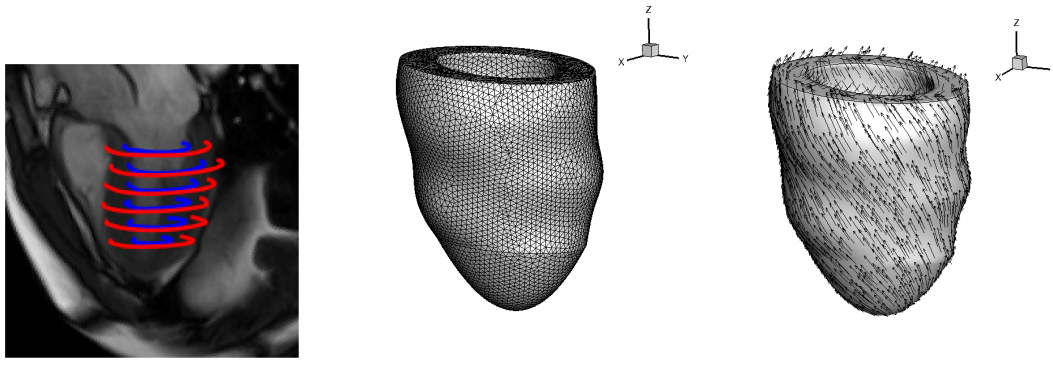
The LV biomechanical model describes the diastolic filling process from early diastole to end diastole. There are many models that can be used to describe this process and these are reviewed in detail in Chabiniok *et al.* (2016). The model that is used here is similar to those used in Wang *et al.* (2013) and Gao *et al.* (2015). The biomechanical model that was initially described in Wang *et al.* (2013) can be thought of as consisting of five parts: initial discretized LV geometry, the constitutive law (the HO law), the constitutive parameters, the finite element implementation, and corresponding initial and boundary conditions. Linking this biomechanical model to patient MRI data can allow the inference of unknown material parameters describing heart mechanics, potentially leading to improved disease diagnosis and personalized treatments (Gao, Mangion, Carrick, Husmeier, Luo and Berry, 2017).

The mathematical model takes three inputs: the initial discretized LV geometry constructed from MRI images at early diastole (Section 2.1), corresponding initial and boundary conditions (Section 2.2) and constitutive parameters (Section 2.3). Based on these inputs, the mathematical model, implemented in ABAQUS (Simulia, Providence, Rhode Island, USA), simulates the diastolic filling process by using the HO law (Section 2.3) and a finite element implementation (Gao *et al.*, 2015). The output of the mathematical model then gives a model of the LV state at end diastole, which can be compared with the corresponding *in vivo* magnetic resonance images. These magnetic resonance images at end diastole are used to measure circumferential strains taken at 24 locations (these are based on the American Heart Association definition as in Gao *et al.* (2015)) and the end diastolic volume. These measurements can be compared against those generated by the biomechanical model for various constitutive parameters to learn the parameters that are associated with the volunteer from whom the magnetic resonance images were taken.

Each simulation from the mathematical model without parallelization takes about 18 min on our local Linux workstation (Intel(R) Xeon(R) central processor unit (CPU), 2.9 GHz, 32 Gbytes memory), or around 4.5 min with parallelization on six CPUs. Note that 18 or 4.5 min are required for just a single parameter adaption step of an iterative optimization, or a single addition to the emulator.

### 2.1. Initial discretized left ventricular geometry

The initial discretized LV geometry can be obtained through constructing a three-dimensional model based on the MRI scans (Wang *et al.*, 2013). The scans consist of a series of six or seven short axis cine images which cover the ventricle. (The MRI study was conducted on a Siemens MAGNETOM Avanto (Erlangen, Germany) 1.5-T scanner with a 12-element phased array cardiac surface coil. Cine MRI images were acquired by using the steady state precession imaging protocol. Patient consent was obtained before the scan.) For each cardiac cycle there are usually around 35 frames from end diastole to early diastole. The images of the early diastole are then used to create the initial discretized LV geometry, whereas the end diastole images will provide the final measurements of the circumferential strains and the LV volume. To create the discretized LV model, the endocardial (inner) and epicardial (outer) boundaries of the left ventricle are segmented from cine images at early diastole as done in Gao, Wang, Berry, Luo




(a) (b) (c)

**Fig. 1.** Biomechanical LV model reconstructed from *in vivo* MRI from a healthy volunteer: (a) segmented ventricular boundaries superimposed on a long axis magnetic resonance image; (b) the reconstructed LV geometry discretized with tetrahedron elements; (c) vector plot of fibre direction **f**, which rotates from endocardium to epicardium

and Griffith (2014), e.g. Fig. 1(a). A three-dimensional model of the left ventricle can then be constructed in Solidworks (Dassault Systems SolidWorks Corp., Waltham, Massachusetts, USA), e.g. Fig. 1(b). Finally, Fig. 1(c) is constructed by using a rule-based fibre generation method (see Gao, Carrick, Berry, Griffith and Luo (2014), giving us the initial discretized LV geometry that was used in the biomechanical model. In the context of the present study, we consider this a fixed input and focus our work on developing parameter inference methods rather than a tool that can work for all possible subjects. Extensions to allow for different LV geometries is the subject of future work.

### *2.2. Initial and boundary conditions*

The initial and boundary conditions, in particular LV pressure, play an important role in myocardial dynamics. Unfortunately, blood pressure within the cavity of the left ventricle can only be measured invasively, by direct catheter measurement within the LV cavity. Because of potential complications and side effects, these measurements are not available for healthy volunteers. We have therefore fixed the boundary conditions, including the pressure, at values that are considered sensible for healthy subjects, based on the work of Bouchard *et al.* (1971).

### *2.3. Constitutive law*

The final part of the biomechanical model is the constitutive law for characterizing the material properties of the myocardium. In this study, we use the invariant-based constitutive law (Holzapfel and Ogden, 2009), based on the following strain energy function:

$$\Psi = \frac{a}{2b}[\exp\{b(I_1-3)\}-1] + \sum_{i\in\{f,s\}} \frac{a_i}{2b_i}[\exp\{b_i(I_{4i}-1)^2\}-1] + \frac{a_{\mathrm{fs}}}{2b_{\mathrm{fs}}}\{\exp(b_{\mathrm{fs}}I_{8\mathrm{fs}}^2)-1\} + \frac{1}{2}K(J-1)^2, \tag{1}$$

in which $a$, $b$, $a_{\mathrm{f}}$, $b_{\mathrm{f}}$, $a_{\mathrm{s}}$, $b_{\mathrm{s}}$, $a_{\mathrm{fs}}$ and $b_{\mathrm{fs}}$ are unknown material parameters, and $I_1$, $I_{4i}$ and $I_{8\mathrm{fs}}$ are the invariants corresponding to the matrix and fibre structure of the myocardium, which are calculated as

$$I_1 = \mathrm{tr}(\mathbb{C}),$$
$$I_{4\mathrm{f}} = \mathbf{f}_0 \cdot (\mathbb{C}\mathbf{f}_0),$$
$$I_{4\mathrm{s}} = \mathbf{s}_0 \cdot (\mathbb{C}\mathbf{s}_0),$$
$$I_{8\mathrm{fs}} = \mathbf{f}_0 \cdot (\mathbb{C}\mathbf{s}_0)$$

in which $\mathbf{f}_0$ and $\mathbf{s}_0$ are the myofibre and sheet orientations, which are determined through a rule-based approach (Wang *et al.*, 2013) and are known before the simulation (initial conditions). $\mathbb{C}$ is the right Cauchy–Green deformation tensor, defined as $\mathbb{C} = \mathbb{F}^{\mathrm{T}}\mathbb{F}$, where $\mathbb{F}$ is the deformation gradient describing the motion of the myocardium and hence how its shape changes in three dimensions with time. The term $\frac{1}{2}K(J-1)^2$ accounts for the incompressibility of the material, where $K$ is a constant ($10^6$) and $J$ is the determinant of $\mathbb{F}$. The HO law forms a major part of the biomechanical model, and the eight constitutive parameters, $a$, $b$, $a_{\mathrm{f}}$, $b_{\mathrm{f}}$, $a_{\mathrm{s}}$, $b_{\mathrm{s}}$, $a_{\mathrm{fs}}$ and $b_{\mathrm{fs}}$, are unknown inputs into the model, which we wish to learn. Assessing the accuracy of parameter estimation for real data can be based on stretch–stress curves, as discussed in section 1 of the on-line supplementary materials.

However, it has previously been found in Gao *et al.* (2015) that the eight parameters are strongly correlated, which suggests that a model reduction is advisable to ensure identifiability. They further demonstrated that myofibre stiffness, which is the parameter that is most relevant for clinical applications, can be estimated from *in vivo* data with a reduced parameterization; see section 2 of the on-line supplementary materials. In fact, Hadjicharalambous *et al.* (2017) even estimated passive myocardial stiffness using a reduced form of the HO law with only a single unknown parameter. In the present study, similarly to Gao *et al.* (2015), we group the eight parameters of equation (1) into four, so that

$$\left.\begin{aligned} a &= \theta_1 a_0, & b &= \theta_1 b_0, \\ a_{\mathrm{f}} &= \theta_2 a_{\mathrm{f}0}, & a_{\mathrm{s}} &= \theta_2 a_{\mathrm{s}0}, \\ b_{\mathrm{f}} &= \theta_3 b_{\mathrm{f}0}, & b_{\mathrm{s}} &= \theta_3 b_{\mathrm{s}0}, \\ a_{\mathrm{fs}} &= \theta_4 a_{\mathrm{fs}0}, & b_{\mathrm{fs}} &= \theta_4 b_{\mathrm{fs}0}, \end{aligned}\right\} \tag{2}$$

where $\theta_i \in [0.1, 5]$, $i = 1, 2, 3, 4$, are the parameters to be inferred from *in vivo* data and $a_0$, $b_0$, $a_{\mathrm{f}0}$, $a_{\mathrm{s}0}$, $b_{\mathrm{f}0}$, $b_{\mathrm{s}0}$, $a_{\mathrm{fs}0}$ and $b_{\mathrm{fs}0}$ are reference values from the published literature (Gao, Aderhold, Mangion, Luo, Husmeier and Berry, 2017). (The reference values are, up to two decimal places, $a_0 = 0.22$, $b_0 = 1.62$, $a_{\mathrm{f}0} = 2.43$, $a_{\mathrm{s}0} = 0.56$, $b_{\mathrm{f}0} = 1.83$, $b_{\mathrm{s}0} = 0.77$, $a_{\mathrm{fs}0} = 0.39$ and $b_{\mathrm{fs}0} = 1.70$.) Our results obtained with this dimension reduction are consistent with the experimental results that were reported in Dokos *et al.* (2002).

## 3. Statistical methodology

This section reviews the notion of a simulator and emulator, as well as establishing the notation that is used throughout the rest of the paper. It also provides details about the emulation strategies that will be used in this paper, as well as the interpolation methods that are considered. The code for our emulation strategies, as well as the simulated data (described in Section 4), is provided at `github.com/vinnydavies/left-ventricle-jrss-c`.

### 3.1. Simulation

A *simulator* $\mathbf{m}$ is a mathematical model that relies on a computationally expensive numerical solution of the underlying system's equations. In the present study, the mathematical model is

the soft tissue mechanical description of the left ventricle based on the HO strain energy function, as discussed in the previous section. The inferential process, i.e. estimating the unknown inputs or parameters $\boldsymbol{\theta}_0$ underlying the observed clinical data $\mathbf{y}_0$, is computationally expensive and infeasible in settings where solutions are required within a short timeframe, for instance in the context of clinical decision support. The prohibitive computational time that makes inference challenging is due to the time that is needed for a single (*forward*) *simulation* from the computational model, where by forward simulation we mean generating a (possibly multivariate) output $\mathbf{y} = (y_1, \ldots, y_J) = \mathbf{m}(\boldsymbol{\theta})$ for a given parameter vector or input $\boldsymbol{\theta}$. In the context of the present study, $J = 25$, and the outputs $y_i$ are the 24 circumferential strains and the LV volume at end diastole, as predicted by the mathematical model.

Given our clinical data, $\mathbf{y}_0$, which are the measured circumferential strains and the end-of-diastole LV volume obtained from MRI, we can estimate the unknown parameter vector $\boldsymbol{\theta}_0$ by finding the corresponding input to the simulator which gives rise to an output which is as close as possible to the observed clinical data $\mathbf{y}_0$. Although our clinical data are assumed to come from the same data-generating process $\mathbf{m}$ for an unknown input $\boldsymbol{\theta}_0$, in practice there will be a systematic deviation due to noisy measurement and model mismatch. A standard approach to estimating the unknown input or parameter vector $\boldsymbol{\theta}$ is to choose the loss function as the negative log-likelihood:

$$l(\boldsymbol{\theta}|\mathbf{m}, \mathbf{y}_0) = \alpha d\{\mathbf{m}(\boldsymbol{\theta}), \mathbf{y}_0\} + Z, \tag{3}$$

for a given metric function $d$ measuring the distance between a simulation $\mathbf{y} = \mathbf{m}(\boldsymbol{\theta})$ and data $\mathbf{y}_0$, and some positive constants $\alpha$ and $Z$. We can then estimate the input to the model by minimizing the true loss (3):

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} l(\boldsymbol{\theta}|\mathbf{m}, \mathbf{y}_0), \tag{4}$$

effectively giving us the maximum likelihood estimate of $\boldsymbol{\theta}$. This method becomes prohibitive if a single simulation exceeds a certain amount of time, as it does with the biomechanical model that is considered in the present work. The numerical procedure based on finite element discretization requires approximately 18 min for a single simulation, or 4.5 min with parallelization on six CPUs on our computing system (a dual Intel Xeon CPU E5-2699 v3, 2.30 GHz, 36 cores and 128 Gbytes memory). Any optimization of the true loss (3) would require the evaluation of the simulator at every iteration of the optimization routine, potentially hundreds or thousands of times, with each iteration taking between 4.5 and 18 min. This is computationally limiting if we wish to use the method for clinical decision support in realtime.

### 3.2. Emulation

An *emulator* is a statistical model that is a cheap and fast approximation to the true computational model (simulator) $\mathbf{m}$, in this case the biomechanical model. It is used to replace the simulator to speed up both computations and inference, and it is also referred to as a *metamodel* (or *surrogate model*) as it represents a model of a model. An emulator can be built by using any interpolation technique such as regression splines, polynomial regression or GPs; see Section 3.4 for more details. Once a method has been chosen and the emulator has been fitted to the training data, we shall denote it as $\hat{\mathbf{m}}$.

To fit a statistical model and to replace the simulator, we need training data from the simulator itself in the form of simulations $\mathcal{D} = \{(\boldsymbol{\theta}_1, \mathbf{y}_1), \ldots, (\boldsymbol{\theta}_N, \mathbf{y}_N)\} = \{\boldsymbol{\Theta}, \mathbf{Y}\}$. In the context of the present application, the input vectors $\boldsymbol{\theta}_i$ are the biomechanical parameter vectors that were

discussed in Section 2.3. These inputs into the simulator, $\boldsymbol{\Theta}$, are chosen on the basis of a space filling design, using Sobol sequences. These so-called low discrepancy sequences are known to lead to improved convergence in the context of quasi-Monte-Carlo algorithms; see for example Gerber and Chopin (2015). A more efficient coverage of the input space is possible by using more advanced statistical design methods, as for example discussed in Overstall and Woods (2017), but these explorations are beyond the remit of the present work.

The outputs of the simulator, $\mathbf{Y}$, are the resulting clinical values based on the assumed data-generating process $\mathbf{m}$. In the present application, the output vectors $\mathbf{y}_i$ are the vectors of 24 circumferential strains and LV volume at end diastole. Although generating large numbers of simulations is computationally expensive, this can be massively parallelized in advance and before the arrival of the patient at the clinic.

Previously, given the clinical data $\mathbf{y}_0$ and a simulator $\mathbf{m}$, we could not estimate the unknown input $\boldsymbol{\theta}_0$ by using the loss function (negative log-likelihood) given in equation (3) sufficiently fast for effective use within a clinical environment. This was due to the high simulation time that is required for each single input. Now, however, we can replace the true loss function (3) with a surrogate loss function $l$ based on an emulation method; see Section 3.3 for details. Minimization of the surrogate loss (surrogate negative log-likelihood) for any metric function $d$ will be fast and suitable for realtime precision medicine, as it does not involve any simulation from the computationally expensive model.

We can use a variety of metric functions within our surrogate loss $l$. The most obvious of these is the Euclidean norm $\|\hat{\mathbf{m}}(\boldsymbol{\theta}) - \mathbf{y}_0\|^2$. Under the assumption of independent and identically normally distributed errors (i.e. deviations of the clinical data from the emulator outputs) with zero mean and variance $\sigma^2$, the Euclidean loss function is equivalent to the negative log-likelihood, up to a scaling factor and an additive constant $Z(\sigma)$:

$$l(\boldsymbol{\theta}|\hat{\mathbf{m}}, \mathbf{y}_0) = \frac{1}{2\sigma^2}\|\hat{\mathbf{m}}(\boldsymbol{\theta}) - \mathbf{y}_0\|^2 + Z(\sigma), \tag{5}$$

where $\hat{\mathbf{m}}$ is the GP predictive mean, which is used to replace the expensive computational model $\mathbf{m}$. An extension of the Euclidean loss which allows for correlations between the outputs is the Mahalanobis loss function

$$l(\boldsymbol{\theta}|\hat{\mathbf{m}}, \mathbf{y}_0) = \tfrac{1}{2}(\hat{\mathbf{m}}(\boldsymbol{\theta}) - \mathbf{y}_0)^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\hat{\mathbf{m}}(\boldsymbol{\theta}) - \mathbf{y}_0) + Z(\boldsymbol{\Sigma}), \tag{6}$$

which is equivalent to the negative log-likelihood of a multivariate Gaussian distribution with covariance matrix $\boldsymbol{\Sigma}$ up to a constant, $Z(\boldsymbol{\Sigma})$. To minimize the computational costs at the clinic, the covariance matrix is precomputed from the training data, $\boldsymbol{\Sigma} = \mathrm{cov}(\mathbf{Y})$, and then kept fixed. Its main purpose is to allow for the spatial correlations between the 24 circumferential strains at different locations on the left ventricle.

### 3.3. Emulation frameworks

#### 3.3.1. Output emulation

Emulating the outputs of the simulator, the LV model, involves fitting multiple individual models, one for each of the $J$ outputs of the simulator $\mathbf{m}$. These outputs, $y_j$, $j = 1, \ldots, J$, are fitted using the inputs of the simulator, $\boldsymbol{\Theta}$, with an appropriate interpolation method; see Sections 3.4.1 and 3.4.2. Given the multiple independent models $\hat{\mathbf{m}} = (\hat{m}_1, \ldots, \hat{m}_J)$, estimates of the parameter vector $\hat{\boldsymbol{\theta}}_0$ can be found for any new set of outputs $\mathbf{y}_0$ by minimizing the difference between $\mathbf{y}_0$ and $\hat{\mathbf{m}}(\boldsymbol{\theta})$ with a loss function:

$$\hat{\boldsymbol{\theta}}_0 = \arg\min_{\boldsymbol{\theta}} l(\boldsymbol{\theta} \mid \hat{\mathbf{m}}, \mathbf{y}_0). \tag{7}$$

The loss function $l$ in equation (7) can take a variety of forms, including the Euclidean and the Mahalanobis loss functions given in equations (5) and (6). An algorithmic description of the output emulation method is given in algorithm 1 in Table 1.

The advantage of emulating the outputs is that the statistical models can be fitted in advance, before the data have been collected from the clinic, meaning that, when a patient comes into the clinic, an estimation of the biomechanical parameter vector $\hat{\boldsymbol{\theta}}_0$ can be carried out relatively quickly. The disadvantage is that multiple potentially correlated model outputs must be fitted, leading to higher computational costs at training time than emulating the loss function directly.

### *3.3.2. Loss emulation*

An alternative strategy is *loss emulation*. This entails direct emulation of the losses $l_n = l(\boldsymbol{\theta}_n|\mathbf{m}, \mathbf{y}_0)$ rather than the simulator outputs $\mathbf{y}_n = \mathbf{m}(\boldsymbol{\theta}_n)$, for $n = 1, \ldots, N$. To follow this approach we fit a single real-valued emulator to training data:

$$\mathcal{D}_l = \{(\boldsymbol{\theta}_n, l_n) : n = 1, \ldots, N\}, \tag{8}$$

where $l_n = l(\boldsymbol{\theta}_n|\mathbf{m}, \mathbf{y}_0)$ is the loss function, for a given metric $d$, evaluated at the $n$th design point from the corresponding simulation output, $\mathbf{y}_n = \mathbf{m}(\boldsymbol{\theta}_n)$. The metric $d$ should be chosen according to the problem, and it can capture the correlation between the model outputs. Now it is possible to fit a single real-valued emulator $\hat{l}(\boldsymbol{\theta}|\mathbf{m}, \mathbf{y}_0)$ of $l(\boldsymbol{\theta}|\mathbf{m}, \mathbf{y}_0)$ based on the training data $\mathcal{D}_l$, using a single statistical model instead of a vector of model outputs. Estimation of the parameters can now be done cheaply by minimizing the emulated loss function:

$$\hat{\boldsymbol{\theta}}_0 = \arg\min_{\boldsymbol{\theta}} \mathbb{E}\{\hat{l}(\boldsymbol{\theta}|\mathbf{m}, \mathbf{y}_0)\}, \tag{9}$$

where $\mathbb{E}$ denotes the conditional expectation predicted by the interpolation method, in our case the conditional mean of a GP. An algorithmic description of the loss emulation method is given in algorithm 2 in Table 2. For further illustration, an additional example, on the Lotka–Volterra system, can be found in Noè (2019).

The advantage of loss emulation over output emulation is a reduction of the training complexity, as a multi-dimensional vector is replaced by a scalar as the target function. The disadvantage is that, as opposed to output emulation, the emulator can only be trained *after* the patient has come into the clinic and the training data have become available. This implies that, on production of the training data, the emulator must be trained and the resulting emulated loss function must be optimized, leading to higher computational costs at the time that a clinical decision must be made. However, these computational costs are still low compared with running the simulator.

Loss emulation is closely related to Bayesian optimization, reviewed for example in Shahriari

**Table 1.** Algorithm 1: inference using an emulator of the outputs

| |
|---|
| *Step 1*: simulate from the model $\mathbf{m}(\boldsymbol{\theta}_1), \ldots, \mathbf{m}(\boldsymbol{\theta}_N)$ at space filling inputs $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_N$ |
| *Step 2*: fit $J$ independent real-valued emulators $\hat{\mathbf{m}} = (\hat{m}_1, \ldots, \hat{m}_J)$, one for each of the $j = 1, \ldots, J$ outputs of the simulator |
| *Step 3*: given data $\mathbf{y}_0$ and the emulator $\hat{\mathbf{m}}$, construct the surrogate-based loss function $l(\boldsymbol{\theta} \mid \hat{\mathbf{m}}, \mathbf{y}_0)$ |
| *Step 4*: minimize the surrogate-based loss function to give the estimates $\hat{\boldsymbol{\theta}}_0$ |

**Table 2.** Algorithm 2: inference using an emulator of the losses

| |
|---|
| *Step 1*: simulate from the model $\mathbf{m}(\boldsymbol{\theta}_1),\ldots,\mathbf{m}(\boldsymbol{\theta}_N)$ at space filling inputs $\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_N$<br>*Step 2*: calculate the set of loss functions $l(\boldsymbol{\theta}_n \mid \mathbf{m}, \mathbf{y}_0)$, for $n = 1,\ldots,N$, between each individual simulation and the observed data $\mathbf{y}_0$<br>*Step 3*: emulate the losses by using a single real-valued model $\hat{l}(\boldsymbol{\theta} \mid \mathbf{m}, \mathbf{y}_0)$<br>*Step 4*: estimate $\hat{\boldsymbol{\theta}}_0$ by minimizing the mean of the loss emulator $\mathbb{E}\{\hat{l}(\boldsymbol{\theta} \mid \mathbf{m}, \mathbf{y}_0)\}$ |

*et al.* (2016) and Noè (2019), which is a strategy to include further query points iteratively by trading off exploration *versus* exploitation via some heuristic or information theoretic criterion. However, every additional query point requires a computationally expensive simulation from the mathematical model, which prevents fast clinical decision making in realtime and renders Bayesian optimization infeasible for the purposes of our study.

### 3.4. Interpolation methods

We have considered several interpolation methods, based on GPs. GPs have been widely used in the context of emulation; see for example Kennedy and O'Hagan (2001), Conti *et al.* (2009) and Conti and O'Hagan (2010). For a comprehensive introduction to GPs, the reader is referred to Rasmussen and Williams (2006). Each of the interpolation methods can be used with both of the emulation paradigms that were described in Section 3.3.

#### 3.4.1. Local Gaussian process

When the sample size $N$ is large, it is not feasible to use exact GP regression on the full data set, because of the $O(N^3)$ computational complexity of the $N \times N$ training covariance matrix inversion. A possible approach is to use sparse GPs as in Titsias (2009), which considers a fixed number of $m$ inducing variables $\mathbf{u} = (u_1,\ldots,u_m)$, with $m \ll N$, corresponding to inputs $\mathbf{Z} = (\mathbf{z}_1,\ldots,\mathbf{z}_m)^{\mathrm{T}}$. The locations of the inducing points and the kernel hyperparameters are chosen with variational inference, i.e. by maximizing a lower bound on the log-marginal-likelihood, which can be derived by applying Jensen's inequality. The computational costs of this approach are $O(Nm^2)$. Initially we tried sparse GPs with 100, 500 and 1000 inducing points but, using the code accompanying the paper by Titsias (2009), the prediction time was between 0.5 and 0.6 s for 100 inducing points, around 1 s for 500 and of the order of a few seconds for 1000 inducing points (dual Intel Xeon CPU E5-2699 v3, 2.30 GHz, 36 cores and 128 Gbytes memory). This means that minimization of the surrogate-based loss would still be slow as approximately 1 s is required for a single evaluation. The optimization time would exceed 2.5 h for 500 inducing points when using 10000 function evaluations. With the cost of variational sparse GP models with larger numbers of inducing points being so large, we can use only about 100 inducing points to keep to our goal of realtime in-clinic decision making. However, using such few inducing points was found to lead to around a quarter of the outputs of the biomechanical model being poorly predicted.

With the performance of the variational sparse GPs being poor when the number of inducing points is selected to give a clinically relevant decision time, we instead use a local GP approach based on the $K$ nearest neighbours instead (Gramacy and Apley, 2015). This method uses the standard GP prediction formulae that were described in Rasmussen and Williams (2006), but subsetting the training data. Whenever we require a prediction at a given input, we find the training inputs representing the $K$ nearest neighbours in input domain, which will form the local set of training inputs, and the corresponding outputs will represent the local training outputs. Note that, every time that we ask for a prediction at a different input, the training sets

need to be recomputed and the GP needs to be trained again. However, because of the small number of neighbours $K \ll 1000$ that were usually selected, this method is computationally fast and accurate; see Gramacy and Apley (2015) for a discussion.

Gramacy and Apley (2015) further discussed adding a fixed number of distant points to help in the estimation of the length scale parameters, but this comes with extra computational costs required by the iterative choice of which point to add to the set of neighbours. Given the time limitations that are required by our goal (realtime clinical decision support systems) we do not pursue this approach. Furthermore, this is mostly relevant when the interest lies in building predictive models that can make good predictions when the training data are distant from each other. Since we are working on a compact set which is densely covered by the Sobol sequence, this is less relevant. For generic training data $\mathcal{D} = \{(\boldsymbol{\theta}_1, y_1), \ldots, (\boldsymbol{\theta}_N, y_N)\} = \{\boldsymbol{\Theta}, \mathbf{y}\}$, we give an algorithmic description in algorithm 3 in Table 3.

In algorithm 3, the $K \times K$ training covariance matrix is $\mathbf{K} = [k(\boldsymbol{\theta}'_i, \boldsymbol{\theta}'_j)]_{i,j=1}^{K}$, the $K \times 1$ vector of covariances between the training points and the test point is $\mathbf{k}(\boldsymbol{\theta}_*) = (k(\boldsymbol{\theta}'_1, \boldsymbol{\theta}_*), \ldots, k(\boldsymbol{\theta}'_K, \boldsymbol{\theta}_*))$ and $\boldsymbol{\mu} = (\mu(\boldsymbol{\theta}'_1), \ldots, \mu(\boldsymbol{\theta}'_K))$ is the $K \times 1$ prior mean vector. We consider a constant mean function $\mu(\boldsymbol{\theta}) = c$. For the kernel $k(\cdot, \cdot)$ we choose the automatic relevance determination squared exponential kernel (see for example Rasmussen and Williams (2006)), as widely used in the emulation of computer codes literature; see for example Fang *et al.* (2006) and Santner *et al.* (2003). The kernel hyperparameters are the output scale (determining the function variance) and the input length scales: one length scale for each dimension. These hyperparameters are estimated by maximizing the log-marginal-likelihood by using the quasi-Newton method. The standard deviation of the additive Gaussian noise $\sigma$ is initialized at a small value, $\sigma = 10^{-2}$, to reflect the fact that the mathematical model of the left ventricle is deterministic. (Even for deterministic models, a small non-zero value for $\sigma$ is usually assumed, to avoid numerical instabilities of the covariance matrix inversion.)

The CPU time that was required to obtain a prediction from the local GP is approximately 0.18 s (dual Intel Xeon CPU E5-2699 v3, 2.30 GHz, 36 cores and 128 Gbytes memory) by using the $K = 100$ nearest neighbours of a given point. The number of neighbours $K$ needs to be selected on the basis of the computational time that is allowed to reach a decision in a viable timeframe, but keeping in mind that $K$ also controls the accuracy of the emulation. In our experiments we found that $K = 100$ was sufficiently fast for the method to be applicable in the clinic while leading to accurate predictions at the test inputs, as discussed below in Section 5.

For this method, the surrogate-based loss and the emulated loss were optimized by using the global search constrained optimization algorithm by Ugray *et al.* (2007), over the bounded domain $[0.1, 5]^4$, which is implemented in MATLAB's Global Optimization toolbox. (Available from `https://uk.mathworks.com/products/global-optimization.html`: we use the default choice of 2000 trial points and 400 stage 1 points. Consider running a local solver from a given starting point $\boldsymbol{\theta}_0$, ending up at the point of local minimum $\hat{\boldsymbol{\theta}}$. The *basin of attraction* corresponding to that minimum is defined as the sphere centred at $\hat{\boldsymbol{\theta}}$

**Table 3.** Algorithm 3: predicting from a local GP at $\boldsymbol{\theta}_*$

*Step 1*: find the indices $\mathcal{N}(\boldsymbol{\theta}_*)$ of the points in $\boldsymbol{\Theta}$ having the $K$ smallest Euclidean distances from $\boldsymbol{\theta}_*$
*Step 2*: training inputs, $\boldsymbol{\Theta}_K(\boldsymbol{\theta}_*) = \{\boldsymbol{\theta}'_1, \ldots, \boldsymbol{\theta}'_K\} = \{\boldsymbol{\theta}_i : i \in \mathcal{N}(\boldsymbol{\theta}_*)\}$
*Step 3*: training outputs, $\mathbf{y}_K(\boldsymbol{\theta}_*) = \{y'_1, \ldots, y'_K\} = \{y_i : i \in \mathcal{N}(\boldsymbol{\theta}_*)\}$
*Step 4*: train a GP by using the data $\mathcal{D}_K(\boldsymbol{\theta}_*) = \{\boldsymbol{\Theta}_K(\boldsymbol{\theta}_*), \mathbf{y}_K(\boldsymbol{\theta}_*)\}$
*Step 5*: predictive mean, $\hat{m}(\boldsymbol{\theta}_*) = \mu(\boldsymbol{\theta}_*) + \mathbf{k}(\boldsymbol{\theta}_*)^{\mathrm{T}}(\mathbf{K} + \sigma^2\mathbf{I})^{-1}(\mathbf{y}_K(\boldsymbol{\theta}_*) - \boldsymbol{\mu})$
*Step 6*: predictive variance, $s^2(\boldsymbol{\theta}_*) = k(\boldsymbol{\theta}_*, \boldsymbol{\theta}_*) - \mathbf{k}(\boldsymbol{\theta}_*)^{\mathrm{T}}(\mathbf{K} + \sigma^2\mathbf{I})^{-1}\mathbf{k}(\boldsymbol{\theta}_*)$

and having radius equal to $\|\boldsymbol{\theta}_0 - \hat{\boldsymbol{\theta}}\|$. All starting points falling inside the sphere are assumed to lead to the same local minimum $\hat{\boldsymbol{\theta}}$; hence no local solver is run and they are discarded. In simple words, stage 1 of the global search algorithm scatters initial points in the domain and scores them from best to worst by evaluating the function value and constraints. Then an interior point local solver (Byrd *et al.*, 2000) is run from each trial point, starting from the point that was scored best (lowest function value), and excluding points that fall into the basins of attraction of previously found minima. When all the stage 1 points have been analysed, stage 2 generates more random points and the same procedure is run a second time.)

### *3.4.2. Low rank Gaussian processes*

Along with local GPs based on the $K$ nearest neighbours, described in Section 3.4.1, we report results for another type of statistical approximation: low rank GPs, as described in section 5.8.2 of Wood (2017), whose main ideas are summarized here for generic training data $\mathcal{D} = \{(\boldsymbol{\theta}_1, y_1), \ldots, (\boldsymbol{\theta}_n, y_n)\} = \{\boldsymbol{\Theta}, \mathbf{y}\}$.

Let $\mathbf{C} = \mathbf{K} + \sigma^2\mathbf{I}$ be the $n \times n$ covariance matrix of $\mathbf{y}$ and consider its eigendecomposition $\mathbf{C} = \mathbf{U}\mathbf{D}\mathbf{U}^{\mathrm{T}}$ with eigenvalues $|D_{i,i}| \geqslant |D_{i+1,i+1}|$. Denote by $\mathbf{U}_k$ the submatrix consisting of the first $k$ eigenvectors of $\mathbf{U}$, corresponding to the top $k$ eigenvalues in $\mathbf{D}$. Similarly, $\mathbf{D}_k$ is the diagonal matrix containing all eigenvalues that are greater than or equal to $D_{k,k}$. Wood (2017) considered replacing $\mathbf{C}$ with the rank $k$ approximation $\mathbf{U}_k\mathbf{D}_k\mathbf{U}_k^{\mathrm{T}}$ obtained from the eigendecomposition. Now, the main issue is how to find $\mathbf{U}_k$ and $\mathbf{D}_k$ sufficiently efficiently. A full eigendecomposition of $\mathbf{C}$ requires $O(N^3)$ operations, which somewhat limits the applicability of the rank reduction approach. A solution is to use the Lanczos iteration method to find $\mathbf{U}_k$ and $\mathbf{D}_k$ at the substantially lower cost of $O(N^2k)$ operations; see section B.11 in Wood (2017). Briefly, the algorithm is an adaptation of power methods to obtain the truncated rank $k$ eigendecomposition of an $N \times N$ symmetric matrix in $O(N^2k)$ operations. However, for large $N$, even $O(N^2k)$ becomes prohibitive. In this scenario the training data are randomly subsampled by keeping $n_r$ inputs and an eigendecomposition is obtained for this random selection with $O(n_r^2k)$ computational cost.

We used the implementation that was found in the R package `mgcv` by Wood (2017), with the following settings: $n_r = 2000$ (the package default), $k = 2000$ for output emulation, and $k = 1000$ for loss emulation. The kernel that was used was an isotropic Matérn 3/2 kernel, with length scale set to the default of Kammann and Wand (2003): $\lambda = \max_{ij} \|\boldsymbol{\theta}_i - \boldsymbol{\theta}_j\|$. The remaining model hyperparameters are estimated by maximizing the log-marginal-likelihood. The final model used an interaction term between each of the four model parameters, as well as a second interactive term between the inverses of the model parameters:

$$\tilde{\mathbf{y}}_j \sim \beta_j\mathbf{1} + f(\boldsymbol{\theta}) + f(\boldsymbol{\tau}) + \boldsymbol{\varepsilon} \qquad \text{for } j = 1, \ldots, J \tag{10}$$

where $\boldsymbol{\tau} = 1/\boldsymbol{\theta}$, $f(\boldsymbol{\theta}) \sim \mathrm{GP}_{\mathrm{LR}}\{\mu(\boldsymbol{\theta}), K(\boldsymbol{\theta}, \boldsymbol{\theta}')\}$, $f(\boldsymbol{\tau}) \sim \mathrm{GP}_{\mathrm{LR}}\{\mu(\boldsymbol{\tau}), K(\boldsymbol{\tau}, \boldsymbol{\tau}')\}$ and $\mathrm{GP}_{\mathrm{LR}}(\cdot)$ denotes a low rank GP. The model specification with the two interaction terms was found to reduce the variation in the predictive accuracy as the volume increases and the strains decrease. This can be seen in the predictions of the test and training data in Figs 2 and 3 of the on-line supplementary materials.

Minimization of the surrogate-based loss $l(\cdot \mid \hat{\mathbf{m}}, \mathbf{y}_0)$ and the emulated loss $\hat{l}(\cdot \mid \mathbf{m}, \mathbf{y}_0)$ was performed by the conjugate gradient method implemented in the R function `optim` (Nash, 1990), with the maximum number of iterations set to 100. To avoid being trapped in local minima, 50 different starting points from a Sobol sequence were used. The best minimum found, subject to not violating the $[0.1, 5]^4$ hyperinterval constraint, was kept as the estimate, discarding the remaining 49 optima.

*3.4.3. Multivariate output Gaussian processes*

The previous two subsections have focused on single-output GPs, while potentially correcting for the correlation structure of the outputs via a modified objective function, using the Mahalanobis distance that is defined in equation (6). We can model the correlation structure between the outputs directly via

$$\text{cov}\{\mathbf{y}(\boldsymbol{\theta}_i), \mathbf{y}(\boldsymbol{\theta}_j)\} = K(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)\mathbf{A} \tag{11}$$

where $K(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)$ is the covariance between $y_k(\boldsymbol{\theta}_i)$ and $y_k(\boldsymbol{\theta}_j)$ for any output $k$, and $\mathbf{A}$ is a matrix of the covariances between the outputs, i.e. the circumferential strains and the LV volume. Various approaches have been proposed in the literature. The approach that was taken in Conti and O'Hagan (2010) and Conti *et al.* (2009) is to place a non-informative prior on $\mathbf{A}$ and to integrate $\mathbf{A}$ out in the likelihood. This leads to a closed form solution in terms of a matrix–normal distribution; see Conti and O'Hagan (2010) and Conti *et al.* (2009) for explicit expressions. However, we found that in combination with algorithm 3—to deal with the $O(N^3)$ computational complexity—the computational costs of running the emulator were of the order of hours, rather than minutes, which renders this approach not viable for clinical decision support in realtime.

An alternative approach is to model the correlation structure of the outputs explicitly via

$$\text{cov}\{y_k(\boldsymbol{\theta}_i), y_l(\boldsymbol{\theta}_j)\} = K(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)A(\mathbf{u}_k, \mathbf{u}_l) \tag{12}$$

taking into account covariates $\mathbf{u}_k$ and $\mathbf{u}_l$ associated with the $k$th and $l$th outputs, $y_k$ and $y_l$ respectively. Roberts *et al.* (2013) pursued this approach in the context of time series analysis, where $\mathbf{u}_k$ and $\mathbf{u}_l$ are scalar variables indicating different time points. In our application, $\mathbf{u}_k$ and $\mathbf{u}_l$ are vectors indicating the locations on the surface of the left ventricle associated with the circumferential strains. Because of the highly non-Euclidean geometry of this space, the choice of kernel is not obvious. A naive approach that we tried is to project the locations onto a linear space defined by the first principal component (Huang, 2016). The results were not encouraging, because of the information loss that was incurred by the map. Future work could try projections onto non-linear maps, like Hilbert curves (Hilbert, 1891; Hamilton and Rau-Chaplin, 2007), generative topographic maps (Bishop *et al.*, 1998) or self-organizing maps (Kohonen, 1982).

A further alternative is the method of Alvarez and Lawrence (2009, 2011), who have proposed sparse convolved GPs for multioutput regression. Their method assumes that there is an underlying process which governs all of the outcomes of the model and treats it as a latent process. Modelling this latent process as a GP leads to a GP prior over the outputs, inducing cross-covariance between the outputs and effectively introducing correlations between them. We can use the interpolation method of Alvarez and Lawrence (2009, 2011) within either of the emulation frameworks that were introduced in Section 3.3. There are, however, problems with doing this: training a convolved GP with $N$ training points requires the inversion of a $DN \times DN$ matrix (where $D = 25$ is the number of outputs) which is currently infeasible with all of the training data ($N = 10000$), even when choosing the number of inducing points by using the method that was proposed in Alvarez and Lawrence (2009, 2011). Instead we can choose a strategy that is similar to that proposed in Section 3.4.1. This again, however, proves to be computationally expensive as fitting a single local emulator requires more than 15 min (Intel Xeon CPU E5-606, 2.13 GHz), without consideration of the computational costs of the subsequent optmization of the LV model parameters. When this is included within either of the emulation methods (algorithms 1 and 2), the time becomes too large for a clinical decision support system, as it is infeasible to make a prediction within a clinically relevant timeframe.

Since the focus of our study is to develop an emulation framework for a clinical decision support system that can work in realtime, we have restricted our analysis to the univariate methods that were described in Sections 3.4.1 and 3.4.2.

## 4. Data and simulations

For training the emulator, we used 10000 parameter vectors generated from a Sobol sequence (Sobol, 1967) in a compact four-dimensional parameter space, with $\theta_1, \ldots, \theta_4 \in [0.1, 5]^4$, where the parameter bounds reflect prior knowledge that was available from Gao *et al.* (2015). The four-dimensional parameter vectors are then transformed to the original eight-dimensional parameter space using transformation (2). The eight-dimensional parameter vectors are then inserted into the HO strain energy function (1). Following the finite element discretization method that was described in Wang *et al.* (2013), the soft tissue mechanical equations are numerically solved to produce a 25-dimensional output vector associated with each parameter vector; these are 24 circumferential strains and the LV volume at end diastole. The Sobol sequence is extended to generate an independent test set of an additional 100 parameter vectors, for which the same procedure is followed to associate them with output vectors of circumferential strains and LV volume. As a real data set, we used 24 circumferential strains and the LV volume at end diastole obtained from the cardiac MRI images of a healthy volunteer, following the procedure that was described in Gao *et al.* (2015).

## 5. Results

To summarize, we have introduced two emulation frameworks which can be used to infer the parameters of the LV biomechanical model; see Sections 3.3.1 and 3.3.2. We have applied these methods with two loss functions, the Mahalanobis loss function and the Euclidean loss function, and two interpolation methods, low rank GPs and local GPs; see Sections 3.4.1 and 3.4.2. Testing each combination of these methods means that there is a total of eight alternative procedures.

We have applied and assessed the proposed methods in a two-pronged approach. Firstly, in Sections 5.1, 5.2, 5.3 and 5.4, we have tested the eight combinations of methods on synthetic data, where the true parameter values of the underlying biomechanical model are known; see the previous section for details on how the training and test data were generated. We compare the methods by using the mean-square error (MSE). The distribution of 100 MSEs is given in Fig. 2 and summarized with the median and the first and third quartiles in Table 4, representing three of Tukey's five-number summary.

Finally, we have applied the method with the best performance in Section 5.5 to clinical data generated from a healthy volunteer's cardiac MRI scan, where we can compare our performance against the gold standard results of Gao *et al.* (2015).

### 5.1. Comparison of interpolation methods

Looking at the two interpolation methods, the local GP method (boxplots 5–8 in Fig. 2) outperforms the low rank GP method (boxplots 1–4 in Fig. 2). The reason for the difference in performance between the two methods is the size of the noise variance that is estimated. With the low rank GP method, a larger noise variance is estimated as the interpolation must fit to the entire data set. The larger variance of the errors is in mismatch with the deterministic nature of the process that we aim to model and is the consequence of the loss of modelling flexibility resulting from the low rank approximation and a potential violation of the stationarity assumption that is intrinsic to our choice of kernel.

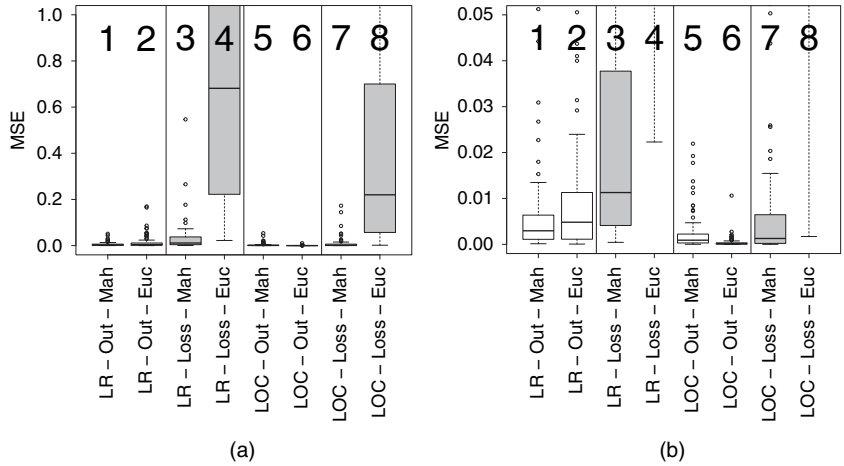



**Fig. 2.** Boxplots of the MSE distribution in the prediction of all the model parameters (the methods from left to right on each plot are as follows: low rank GP (LR) output emulation (Out) with Mahalanobis loss function (Mah) and Euclidean loss function (Euc), LR–GP loss emulation (Loss) with Mahalanobis loss function and Euclidean loss function, local GP (LOC) output emulation with Mahalanobis loss function and Euclidean loss function, and LOC loss emulation with Mahalanobis loss function and Euclidean loss function; the outliers are due to non-convergence of the optimization algorithm and the strong correlation between the parameters of the HO law): (a) boxplots of the MSE in parameter space for all the eight methods; (b) the same boxplots but with a reduced scale on the $y$-axis

**Table 4.** Median (first quartile, third quartile) of the MSE (in parameter space) in the prediction of all the model parameters†

| *Interpolation method* | *Emulation target* | *Results for Euclidean loss function* | *Results for Mahalanobis loss function* |
|---|---|---|---|
| Low Rank GP | Output | 0.0048 (0.0012,0.0107) | 0.0030 (0.0011,0.0062) |
| Low Rank GP | Loss | 0.6814 (0.2222,1.5234) | 0.0113 (0.0041,0.0377) |
| Local GP | Output | *0.0001 (0.0000,0.0003)* | 0.0009 (0.0003,0.0022) |
| Local GP | Loss | 0.2201 (0.0588,0.6777) | 0.0013 (0.0002,0.0063) |

†The interpolation methods considered are low rank GPs and local GPs; the target of the emulation is either the model output or the loss, and two loss functions are compared, Euclidean and Mahalanobis. The method with the best predictive performance, the output emulation method with local GP interpolation and the Euclidean loss function, is given in italics.

Conversely, with the local GP method, a much smaller error variance is estimated, which more closely matches the deterministic data generation method. This is a result of there being only a small number of points that the interpolant must fit. These points are local, giving more detail of the local surface than the low rank GP method, which uses all the points but takes a rank $k$ eigendecomposition of the kernel matrix. The local GP method also provides a natural way to accommodate non-stationary processes.

### *5.2. Comparison of emulation frameworks*

Out of the two emulation frameworks, the output emulation method (boxplots 1, 2, 5 and 6 in

Fig. 2) gives the most accurate parameter estimates, outperforming the loss emulation method (boxplots 3, 4, 7 and 8 in Fig. 2) for all interpolation methods and loss functions. The output emulation method provides accurate estimates for all the combinations of interpolation methods and loss functions, whereas the loss emulation method provides poor estimates in some cases. The improved parameter estimation of the output emulation method is a result of using multiple separate emulators. These multiple emulators better model the complex non-linear relationships between the parameters and the outputs than is possible with the single emulator that is used with the loss emulation method. In the loss emulation method, the differences between the patient data and the simulations are summarized in one loss function, which entails a larger loss of information.

### 5.3. Comparison of loss functions

In terms of the accuracy of the parameter inference, the Euclidean loss and Mahalanobis loss perform differently in different emulation methods. For the loss emulation method the Mahalanobis loss function (boxplots 3 and 7 in Fig. 2) clearly outperforms the Euclidean loss function (boxplots 4 and 8 in Fig. 2) in all cases. The reason for the difference is that the loss function summarizes how similar the patient data are to the simulations and this is done more realistically by the Mahalanobis loss function in this case. This is because there are spatial correlations between the outputs due to measuring the circumferential strains at different neighbouring locations on the left ventricle. The Mahalanobis loss function accounts for this through including a correlation estimate, whereas the Euclidean loss function does not.

In comparison with the loss emulation method, for the output emulation method it is less clear which loss function gives the best results. The Mahalanobis loss function is marginally better for the low rank GP method (boxplot 1 is better than boxplot 2 in Fig. 2), whereas the Euclidean loss function gives the best performance for the local GP method (boxplot 6 is better than boxplot 5 in Fig. 2). The reason why the Euclidean loss function performs best for the local GP method is presumably because of potential inaccuracies in the covariance matrix that is used for the Mahalanobis loss function. The covariance matrix is a global measure based on the whole data set and may not accurately represent the true correlations between the local points because of limited numerical precision. (Using a local covariance matrix was also tested, but limited accuracy and numerical stability of the covariance matrix due to using only a small number of local points meant that the performance did not improve over the global covariance matrix.) This is potentially aggravated by a lack of numerical stability when inverting the covariance matrix.

### 5.4. Overall best method in simulation study

In conclusion, the results of our simulation study show the following results.

(a) The local GP method outperforms the low rank GP method and is the better of the two interpolation methods.
(b) The best emulation method is the output emulation method and this outperforms the loss emulation method in all the combinations of interpolation method and loss function tested.
(c) The Mahalanobis loss function gives the best performance for the loss emulation method.
(d) For the output emulation method, the Mahalanobis method is marginally better for the low rank GP method, but for the local GP method the Euclidean loss function gives the best parameter estimates.
(e) Overall, the simulation study results show that the best performing combination of meth-

ods is the output emulation method, using the local GP as the interpolation method and the Euclidean loss function (boxplot 6 in Fig. 2).

This combination of methods will be used on the cardiac MRI data of the healthy volunteer in Section 5.5.

### *5.5. Application to cardiac magnetic resonance imaging data*

Fig. 2 and Table 4 show that the method which gives the most accurate parameter prediction is the emulation of the outputs method combined with the local GP interpolation and the Euclidean loss function. We have applied this strategy to estimate the material parameters for the heart model of a healthy volunteer described in Section 2, using the set of 24 circumferential strains and the LV cavity volume extracted from cardiac MRI images, as described in Section 4. The true model parameters are not known in this case, so as opposed to the simulation study we do not have a proper 'gold standard' for evaluation. We therefore use the following alternative procedure. We first estimate the constitutive parameters with the method of Gao *et al.* (2015) and Gao, Aderhold, Mangion, Luo, Husmeier and Berry (2017), i.e. with the method using the computationally expensive simulator. From these parameters, we calculate the stretch–stress relationships along the directions of the sheets and the myocytes, following the procedure that was described in Holzapfel and Ogden (2009). We use these graphs as a surrogate gold standard, which we compare with the corresponding graphs obtained from the parameters that were obtained with our emulation approach.

Fig. 3 shows, as broken curves, the estimate of the stretch–stress relationship for the healthy volunteer by using the gold standard method of Gao *et al.* (2015) and Gao, Aderhold, Mangion, Luo, Husmeier and Berry (2017). For comparison, the full curves show the estimates of the stress–stretch relationship that was obtained from the best emulation method identified in Sections 5.1–5.4, the emulation of the outputs method combined with the local GP interpolation method and the Euclidean loss function.

For uncertainty quantification, we numerically estimated the Hessian at the minimum surrogate loss (5). Its inverse represents an approximate lower bound on the variance–covariance matrix in parameter space. (The Hessian is the empirical Fisher information matrix. The lower bound would be exact (Cramer–Rao lower bound) if we could take an expectation with respect to the data distribution. Recall that saying that matrix $\mathbf{A}$ is a lower bound on matrix $\mathbf{B}$ means that $\mathbf{B} - \mathbf{A}$ is positive semidefinite.) The uncertainty in the estimate can then be obtained by sampling from a multivariate normal distribution, with the covariance set to the inverse of the Hessian, $\mathrm{MVN}\{\hat{\boldsymbol{\theta}}, \mathbf{H}(\hat{\boldsymbol{\theta}})^{-1}\}$, and calculating the corresponding confidence intervals.

The results in Fig. 3 show that the emulation method accurately estimates the stretch–stress relationship in the myocyte direction. The agreement between the gold standard and the prediction with our emulation method is nearly perfect, with a deviation that is less than the predicted single-standard deviation width. For the stretch–stress relationship in the sheet direction, the agreement is also very good, although the deviation exceeds the predicted standard deviation in this case. A possible explanation is that parameter sensitivity in the sheet directions is very low when only using regional circumferential strains and the LV cavity volume to formulate the objective function, as reported in Gao *et al.* (2015); thus the uncertainty of estimating the stiffness in the sheet direction will be higher than that in the myocyte direction. It is expected that higher accuracy will be achieved when radial (transmural) strains are included when inferring the parameters. Although the differences between the stretch–stress curves that were obtained with the simulator and our emulator are minor, there is a substantial difference in the computational costs. For the simulator, i.e. the original procedure that was described in Gao *et al.* (2015) and Gao, Aderhold,

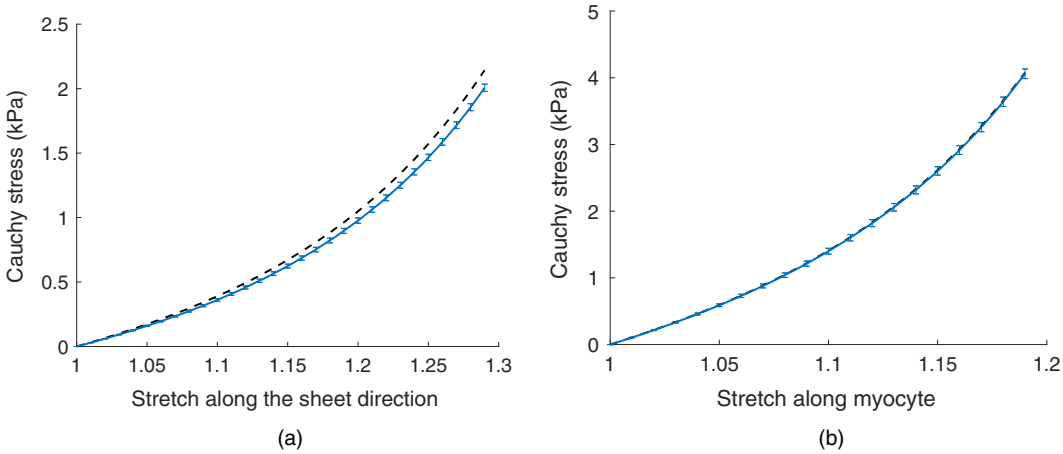



**Fig. 3.** Plots of the Cauchy stress against the stretch along (a) the sheet direction and (b) the myocyte direction: – – –, literature curves taken from the gold standard method in Gao, Aderhold, Mangion, Luo, Husmeier and Berry (2017); ———, estimates of the curves from the best emulation method, the emulation of the outputs method combined with the local GP interpolation method and the Euclidean loss function; ⌶, 95% confidence intervals, approximated by using the sampling method described in Section 5.5

Mangion, Luo, Husmeier and Berry (2017), the computational costs are of the order of over a week. The estimation procedure with the emulator proposed, in contrast, could be carried out in less than 15 min (dual Intel Xeon CPU E5-2699 v3, 2.30 GHz, 36 cores and 128 Gbytes memory), giving us a reduction in the computational complexity by about three orders of magnitude.

Hence, whereas the former procedure is only of interest in a pure research context, the latter procedure gives us estimation times that are acceptable in a clinical decision context. This is an important first step towards bringing mathematical modelling into the clinic and making a real impact in healthcare.

## 6. Discussion

We have developed an emulation framework that can be used to infer the material properties of the left ventricle of a healthy patient in a clinically viable timeframe. We have focused on developing an emulation framework that can be used in future more generalized work and have therefore tested two emulation methods, two interpolation methods and two loss functions; see Section 3. Each combination of these methods has then been evaluated in a simulation study to determine the best method. The best method was found to be the output emulation method, using the local GP as the interpolation method and the Euclidean loss function; see Table 4.

We have then applied the proposed emulation method to cardiac MRI data and demonstrated that it can accurately estimate the stretch–stress relationship along the myocyte and sheet directions of the left ventricle from a healthy volunteer. Our method provides a notable improvement in computational time with a speed-up of approximately three orders of magnitude. In particular, whereas conventional parameter estimation based on numerical simulations from the mathematical LV model, following for example the approach of Gao *et al.* (2015), leads to computational costs of the order of weeks, the proposed emulation method reduces the computational complexity to the order of a quarter of an hour, while effectively maintaining the same level of accuracy. This is an important step towards a clinical decision support system that can assist a clinical practitioner in realtime.

A limitation of the current approach is the fact that the LV geometry is fixed. This LV geometry varies from patient to patient, and these variations need to be taken into consideration for applications to wider patient populations. We discuss how potentially to address this challenge in the next section.

## 7. Future work

The next step for this work is to design a method that is capable of fast parameter inference for multiple patients on whom we have not directly trained the emulator. For each new patient we would need to replace the single geometry that is used here as an input, with the new patient's data on arrival at the clinic. With no time limits on the inference, we could simply replicate this study with a different input geometry. However, to treat patients in a clinically viable timeframe we must be able to train the emulator for the unobserved patients before they enter the clinic. We can do this by using simulations from multiple LV geometries as our training data. Low dimensional representations of each geometry can then be included as variables in the interpolation method of the emulator and we can learn how these changes affect the output of the biomechanical model. When new patient data then arrive, these low dimensional representations can be calculated and included in the loss function, which must be minimized in the emulation method.

A straightforward approach for achieving this low dimensional representation is principle component analysis (PCA), illustrated in Fig. 4(a), where the high dimensional LV geometries are mapped onto a low dimensional space that captures the maximum variation in the population. A variation along the PCA directions can be mapped back into the high dimensional LV geometry space to illustrate typical organ deformations, as illustrated in Fig. 4(b). However, although fast and easy to implement, the limitation of PCA is its restriction to linear subspaces. If the LV geometries that are extracted from the patient population are grouped along a non-linear submanifold in the high dimensional LV geometry space, as illustrated in Fig. 4(c), PCA is suboptimal. A variety of non-linear extensions of and alternatives to PCA have been proposed in the machine learning and computational statistics literature. The most straightforward exten-

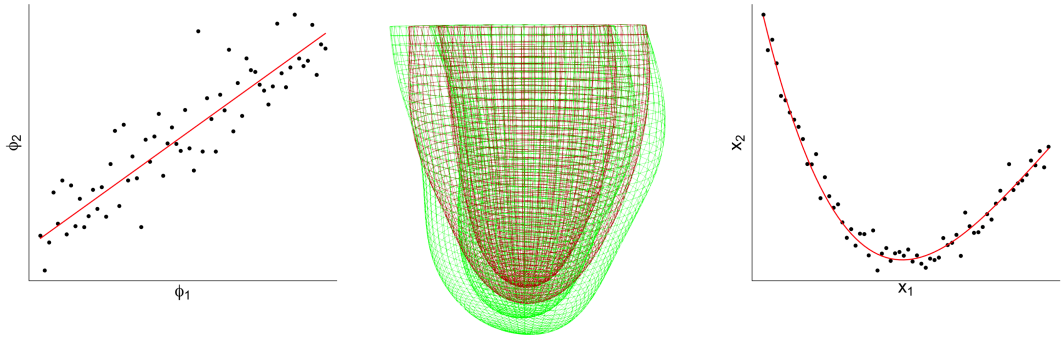



**Fig. 4.** Illustration of dimension reduction for the representation of the left ventricle: (a) illustration of PCA (a set of LV geometries extracted from a set of patients forms a cloud of vectors in a high dimensional vector space (here reduced to 2 for visual representation); PCA provides a set of linear orthogonal subspaces along the directions of maximum variance (here only one, the leading component, is shown)); (b) a variation along the principal component can be mapped back into the high dimensional vector space to show the corresponding changes of the LV geometry (here indicated by different colour shadings); (c) PCA is a linear technique and hence suboptimal if the LV geometries from the patient population are grouped along a non-linear submanifold

sion is kernel PCA (Scholkopf *et al.*, 1998), which conceptually maps the data non-linearly into a high dimensional vector space and makes use of Mercer's theorem, whereby the scalar product in this high dimensional space is equivalent to a kernel in the original data space and therefore never has to be computed explicitly. Alternative non-linear dimension reduction methods to be explored are generative topographic maps (Bishop *et al.*, 1998), self-organizing maps (Kohonen, 1982) and variational autoencoding neural networks (Kingma and Welling, 2014).

## 8. Software

The software developed for and used in our study can be downloaded from `https://github.com/vinnydavies/left-ventricle-jrss-c`.

## Acknowledgements

This work was funded by the British Heart Foundation, grants PG/14/64/31043 and RE/18/6134 217, and by the UK Engineering and Physical Sciences Research Council, grant EP/N014642/1, as part of the SofTMech project. Benn Macdonald is supported by the Biometrika Trust, Fellowship B0003. Umberto Noè was supported by a Biometrika Research Studentship. Alan Lazarus is partially funded by a grant from GlaxoSmithKline plc. Dirk Husmeier is supported by a grant from the Royal Society of Edinburgh, award 62335.

*Author contribution*

Hao Gao and Xiaoyu Luo extracted the LV geometry from the MRI scans and integrated it into the mathematical model that is described in Section 2. Colin Berry obtained and contributed the MRI data. Benn Macdonald ran the parallelized simulations from the model for training the emulator. Vinny Davies developed the low rank GP emulator. Umberto Noè developed the local GP emulator. Alan Lazarus investigated the multivariate output GP approach. Dirk Husmeier led and co-ordinated the statistical emulation and inference work. All authors discussed the results and contributed to writing the paper.

Vinny Davies and Umberto Noè contributed equally to the paper.

## References

Achille, P. D., Harouni, A., Khamzin, S., Solovyova, O., Rice, J. J. and Gurev, V. (2018) Gaussian process regressions for inverse problems and parameter searches in models of ventricular mechanics. *Front. Physiol.*

Alvarez, M. and Lawrence, N. D. (2009) Sparse convolved Gaussian processes for multi-output regression. In *Advances in Neural Information Processing Systems 21* (eds D. Koller, D. Schuurmans, Y. Bengio and L. Bottou), pp. 57–64. Neural Information Processing Systems Foundation.

Alvarez, M. A. and Lawrence, N. D. (2011) Computationally efficient convolved multiple output Gaussian processes. *J. Mach. Learn. Res.*, **12**, 1459–1500.

Bishop, C. M., Svensen, M. and Williams, C. K. (1998) GTM: the generative topographic map. *Neurl Computn*, **10**, 215–234.

Bouchard, R. J., Gault, J. H. and Ross, Jr, J. (1971) Evaluation of pulmonary arterial end-diastolic pressure as an estimate of left ventricular end-diastolic pressure in patients with normal and abnormal left ventricular performance. *Circulation*, **44**, 1072–1079.

Byrd, R. H., Gilbert, J. C. and Nocedal, J. (2000) A trust region method based on interior point techniques for nonlinear programming. *J. Math. Program.*, **89**, 149–185.

Chabiniok, R., Wang, V. Y., Hadjicharalambous, M., Asner, L., Lee, J., Sermesant, M., Kuhl, E., Young, A. A., Moireau, P., Nash, M. P., Chapelle, D. and Nordsletten, D. A. (2016) Multiphysics and multiscale modelling, data–model fusion and integration of organ physiology in the clinic: ventricular cardiac mechanics. *Interfc. Foc.*, **6**, article 0083.

Conti, S., Gosling, J., Oakley, J. E. and O'Hagan, A. (2009) Gaussian process emulation of dynamic computer codes. *Biometrika*, **96**, 663–676.

Conti, S. and O'Hagan, A. (2010) Bayesian emulation of complex multi-output and dynamic computer models. *J. Statist. Planng Inf.*, **140**, 640–651.
Dokos, S., Smaill, B. H., Young, A. A. and LeGrice, I. J. (2002) Shear properties of passive ventricular myocardium. *Am. J. Physiol. Hrt Circultry Physiol.*, **283**, H2650–H2659.
Fang, K., Li, R. and Sudjianto, A. (2006) *Design and Modeling for Computer Experiments*. Boca Raton: Chapman and Hall–CRC.
Gao, H., Aderhold, A., Mangion, K., Luo, X., Husmeier, D. and Berry, C. (2017) Changes and classification in myocardial contractile function in the left ventricle following acute myocardial infarction. *J. R. Soc. Interfc.*, **14**, article 0203.
Gao, H., Carrick, D., Berry, C., Griffith, B. E. and Luo, X. (2014) Dynamic finite-strain modelling of the human left ventricle in health and disease using an immersed boundary-finite element method. *IMA J. Appl. Math.*, **79**, 978–1010.
Gao, H., Li, W., Cai, L., Berry, C. and Luo, X. (2015) Parameter estimation in a Holzapfel–Ogden law for healthy myocardium. *J. Engng Math.*, **95**, 231–248.
Gao, H., Mangion, K., Carrick, D., Husmeier, D., Luo, X. and Berry, C. (2017) Estimating prognosis in patients with acute myocardial infarction using personalized computational heart models. *Scient. Rep.*, **7**, 1–14.
Gao, H., Wang, H., Berry, C., Luo, X. and Griffith, B. E. (2014) Quasi-static image-based immersed boundary-finite element model of left ventricle under diastolic loading. *Int. J. Numer. Meth. Biomed. Engng*, **30**, 1199–1222.
Gerber, M. and Chopin, N. (2015) Sequential quasi Monte Carlo (with discussion). *J. R. Statist. Soc.* B, **77**, 509–579.
Gramacy, R. B. and Apley, D. W. (2015) Local Gaussian process approximation for large computer experiments. *J. Computnl Graph. Statist.*, **24**, 561–578.
Guccione, J. M., McCulloch, A. D. and Waldman, L. (1991) Passive material properties of intact ventricular myocardium determined from a cylindrical model. *J. Biomech. Engng*, **113**, 42–55.
Hadjicharalambous, M., Asner, L., Chabiniok, R., Sammut, E., Wong, J., Peressutti, D., Kerfoot, E., King, A., Lee, J., Razavi, R., Smith, N., Carr-White, G. and Nordsletten, D. (2017) Non-invasive model-based assessment of passive left-ventricular myocardial stiffness in healthy subjects and in patients with non-ischemic dilated cardiomyopathy. *Ann. Biomed. Engng*, **45**, 605–618.
Hamilton, C. and Rau-Chaplin, A. (2007) Compact Hilbert indices: space-filling curves for domains with unequal side lengths. *Inform. Process. Lett.*, **105**, 155–163.
Hilbert, D. (1891) Über die stetige Abbildung einer Linie auf ein Flächenstück. *Math. Ann.*, **38**, 459–460.
Holzapfel, G. A. and Ogden, R. W. (2009) Constitutive modelling of passive myocardium: a structurally based framework for material characterization. *Phil. Trans. R. Soc.* A, **367**, 3445–3475.
Huang, Y. (2016) Multivariate adaptive regression splines based emulation of the heart kinematics. *MSc Thesis*. University of Glasgow, Glasgow.
Kammann, E. E. and Wand, M. P. (2003) Geoadditive models. *Appl. Statist.*, **52**, 1–18.
Kennedy, M. C. and O'Hagan, A. (2001) Bayesian calibration of computer models (with discussion). *J. R. Statist. Soc.* B, **63**, 425–464.
Kingma, D. and Welling, M. (2014) Auto-encoding variational Bayes. *Int. Conf. Learning Representations*.
Kohonen, T. (1982) Self-organized formation of topologically correct feature maps. *Biol. Cyber.*, **43**, 59–69.
Mahalanobis, P. C. (1936) On the generalized distance in statistics. *Proc. Natn. Inst. Sci. Calc.*, **2**, 49–55.
Melis, A., Clayton, R. H. and Marzo, A. (2017) Bayesian sensitivity analysis of a 1D vascular model with Gaussian process emulators. *Int. J. Numer. Meth. Biomech. Engng*, **33**, article e2882.
Nash, J. C. (1990) *Compact Numerical Methods for Computers: Linear Algebra and Function Minimisation*. Boca Raton: CRC Press.
Nikou, A., Dorsey, S. M., McGarvey, J. R., Gorman, J. H., Burdick, J. A., Pilla, J. J., Gorman, R. C. and Wenk, J. F. (2016) Computational modeling of healthy myocardium in diastole. *Ann. Biomed. Engng*, **44**, 980–992.
Noè, U. (2019) Bayesian nonparametric inference in mechanistic models of complex biological systems. *PhD Thesis*. University of Glasgow, Glasgow.
Noè, U., Chen, W., Filippone, M., Hill, N. and Husmeier, D. (2017) Inference in a partial differential equations model of pulmonary arterial and venous blood circulation using statistical emulation. In *Computational Intelligence Methods for Bioinformatics and Biostatistics* (eds A. Bracciali, G. Caravagna, D. Gilbert and R. Tagliaferri), pp. 184–198. Cham: Springer.
Overstall, A. M. and Woods, D. C. (2017) Bayesian design of experiments using approximate coordinate exchange. *Technometrics*, **59**, 458–470.
Rasmussen, C. E. and Williams, C. K. I. (2006) *Gaussian Processes for Machine Learning*. Cambridge: MIT Press.
Remme, E. W., Hunter, P. J., Smiseth, O., Stevens, C., Rabben, S. I., Skulstad, H. and Angelsen, B. (2004) Development of an in vivo method for determining material properties of passive myocardium. *J. Biomech.*, **37**, 669–678.
Roberts, S., Osborne, M., Ebden, M., Reece, S., Gibson, N. and Aigrain, S. (2013) Gaussian processes for time-series modelling. *Phil. Trans. R. Soc.* A, **371**, article 0550.
Santner, T. J., Williams, B. J. and Notz, W. I. (2003) *The Design and Analysis of Computer Experiments*. New York: Springer.

Scholkopf, B., Smola, A. and Muller and K. R. (1998) Nonlinear component analysis as a kernel eigenvalue problem. *Neurl Computn*, **10**, 1299–1319.
Sermesant, M., Moireau, P., Camara, O., Sainte-Marie, J., Andriantsimiavona, R., Cimrman, R., Hill, D. L., Chapelle, D. and Razavi, R. (2006) Cardiac function estimation from MRI using a heart model and data assimilation: advances and difficulties. *Med. Im. Anal.*, **10**, 642–656.
Shahriari, B., Swersky, K., Wang, Z., Adams, R. P. and de Freitas, N. (2016) Taking the human out of the loop: a review of Bayesian optimization. *Proc. IEEE*, **104**, 148–175.
Sobol, I. M. (1967) On the distribution of points in a cube and the approximate evaluation of integrals. *Z. Vych. Mat. Mat. Fiz.*, **7**, 784–802.
Sun, K., Stander, N., Jhun, C. S., Zhang, Z., Suzuki, T., Wang, G. Y., Saeed, M., Wallace, A. W., Tseng, E. E., Baker, A. J., Saloner, D., Einstein, D. R., Ratcliffe, M. B. and Guccione, J. M. (2009) A computationally efficient formal optimization of regional myocardial contractility in a sheep with left ventricular aneurysm. *J. Biomech. Engng.*, **131**, article 111001.
Titsias, M. (2009) Variational learning of inducing variables in sparse Gaussian processes. In *Proc. 12th Int. Conf. Artificial Intelligence and Statistics, Clearwater Beach* (eds D. van Dyk and M. Welling), pp. 567–574. New York: Association for Computing Machinery.
Ugray, Z., Lasdon, L., Plummer, J., Glover, F., Kelly, J. and Martí, R. (2007) Scatter search and local NLP solvers: a multistart framework for global optimization. *INFORMS J. Comput.*, **19**, 328–340.
Wang, H. M., Gao, H., Luo, X. Y., Berry, C., Griffith, B. E., Ogden, R. W. and Wang, T. J. (2013) Structure based finite strain modelling of the human left ventricle in diastole. *Int. J. Numer. Meth. Biomed. Engng*, **29**, 83–103.
Wang, V., Nielsen, P. and Nash, M. (2015) Image-based predictive modeling of heart mechanics. *A. Rev. Biomed. Engng*, **17**, 351–383.
Widmaier, E. P., Hershel, R. and Strang, K. T. (2016) *Vander's Human Physiology: the Mechanisms of Body Function*, 14th edn. New York: McGraw-Hill Education.
Wood, S. N. (2003) Thin plate regression splines. *J. R. Statist. Soc.* B, **65**, 95–114.
Wood, S. N. (2017) *Generalized Additive Models: an Introduction with R*. Boca Raton: CRC Press.
Xi, J., Lamata, P., Lee, J., Moireau, P., Chapelle, D. and Smith, N. (2011) Myocardial transversely isotropic material parameter estimation from in-silico measurements based on a reduced-order unscented Kalman filter. *J. Mech. Behav. Biomed. Mater.*, **4**, 1090–1102.
Xi, J., Shi, W., Rueckert, D., Razavi, R., Smith, N. P. and Lamata, P. (2014) Understanding the need of ventricular pressure for the estimation of diastolic biomarkers. *Biomech. Modlng Mechbiol.*, **13**, 747–757.

*Supporting information*

Additional 'supporting information' may be found in the on-line version of this article:

'Fast parameter inference in a biomechanical model of the left ventricle using statistical emulation'.