Classification framework for partially observed dynamical systems

Yuan Shen[*]
School of Computer Science, The University of Birmingham, Birmingham, United Kingdom
and Department of Mathematical Sciences, Xi'an Jiaotong-Liverpool University, Suzhou, China

Peter Tino[†]
School of Computer Science, The University of Birmingham, Birmingham, United Kingdom

Krasimira Tsaneva-Atanasova[‡]
Department of Mathematics, College of Engineering, Mathematics and Physical Sciences, University of Exeter, Exeter EX4 4QF, United Kingdom
and EPSRC Centre for Predictive Modelling in Healthcare, University of Exeter, Exeter EX4 4QF, United Kingdom
(Received 8 September 2016; published 14 April 2017)

We present a general framework for classifying partially observed dynamical systems based on the idea of learning in the model space. In contrast to the existing approaches using point estimates of model parameters to represent individual data items, we employ posterior distributions over model parameters, thus taking into account in a principled manner the uncertainty due to both the generative (observational and/or dynamic noise) and observation (sampling in time) processes. We evaluate the framework on two test beds: a biological pathway model and a stochastic double-well system. Crucially, we show that the classification performance is not impaired when the model structure used for inferring posterior distributions is much more simple than the observation-generating model structure, provided the reduced-complexity inferential model structure captures the essential characteristics needed for the given classification task.

DOI: 10.1103/PhysRevE.95.043303

I. INTRODUCTION

Classification is a basic machine learning task. Conventional classification algorithms operate on numerical vectors. Over the past decade, such algorithms have been extended for classifying data with more complex structure, e.g., time series data [1,2]. In many real-world applications, time series data can be irregularly and/or sparsely sampled. This poses a challenge for time series classification. On the other hand, the data-generating processes in such applications could be well understood and mechanistic models accounting for the data structure could have been developed in the form of dynamical systems. Using such mechanistic models in time series classification would allow for natural incorporation of the domain experts' knowledge. In this setting, time series data can be seen as partial observations of the underlying dynamical system and the machine learning task becomes classification of partially observed dynamical systems. In this work, we formulate and validate a general framework for such classification tasks.

Xing *et al.* [2] distinguish two major conventional approaches to time series classification, in particular, feature-based and distance-based approaches. Feature based approaches construct discriminative features on the time series data. These can be local patterns (i.e., short subsequences) [3], or global ones resulting from time-frequency and wavelet analysis [4]. Distance-based methods classify time series based on a distance (e.g., Euclidean distance) between time series pairs. This approach is not directly applicable for the time series of variable length. To circumvent this problem, "dynamical time wrapping" (DTW) methods have been developed. In DTW two time series are aligned according to some criteria so that a distance can be calculated [5]. However, such approaches are not applicable for classifying irregularly and sparsely sampled time series. More importantly, they do not utilize the available experts' knowledge about the underlying processes. Alternatively, model-based approaches have also been adopted for time series classification, e.g., hidden Markov model (HMM)–based approaches for biological sequence classification [6]. In those approaches, a prototypical time series model is constructed for each time series class. For example, if the prototypical model is probabilistic, the class label for a new time series is given by the model with the highest likelihood for that time series (or the highest posterior probability, if class priors are available). However, a single model may not adequately represent all time series in the given class. From this point of view, it is more desirable to represent time series by individual models. In this setting, the classifier employed classifies individual models (that stand for individual time series) and thus operates in the model space. We refer to this approach as "learning in the model space" (LiMS) and have adopted it for classifying partially observed dynamical systems.

In most LiMS methods for time series classification, given a time series, a point estimate of model parameter is used to represent that time series. Such estimates could be used directly as feature vectors. In this case, any vector-based classifier could be employed for the task. For example, Brodersen *et al.* [7] employ the dynamic causal model (DCM) [8] to represent individual functional magnetic resonance imaging

[*]Corresponding author: y.shen.2@cs.bham.ac.uk; Yuan.Shen@xjtlu.edu.cn
[†]pxt@cs.bham.ac.uk
[‡]K.Tsaneva-Atanasova@exeter.ac.uk

(fMRI) data from each participant. The maximum–*a posteriori* estimates of a model parameter were then used as feature vectors for classifying DCMs. In Chen *et al.* [9,10], a reservoir computation model was used as a generic nonparametric model to represent nonlinear time series data. A high dimensional dynamical reservoir was fixed and individual time series were represented by the corresponding read-out mappings from the generic dynamic reservoir. The estimated read-out parameters were then used as a feature vector for time series classification. In both approaches, their respective parameter space is considered as a linear metric space and its global metric tensor can be learned in a supervised manner, so as to improve the classification performance.

Other LiMS approaches use direct model distances (e.g., geodesic on the model manifold) instead of global metric in the parameter space. Such approaches treat the parameter space as a nonlinear metric space and learn metric on the underlying manifold. Such a nonlinear structure could be induced by the intrinsic properties of the underlying processes, or by the constraints imposed on the models [e.g., stability of autoregressive (AR) models]. To compute geodesic distances, one can first reconstruct the underlying metric tensor field in the parameter space. Cuzzolin [11] and Cuzzolin and Sapienza [12] propose a general framework based on pullback metric to learn discriminative metric tensors in the space of linear dynamical systems (LDSs) and hidden Markov models (HMMs), respectively. The manifold structure in the parameter space is induced by stability constraints on the LDS parameters, or by normalization constraints on the HMM parameters.

Yet another class of LiMS approaches is formulated in the framework of kernel machines. Although the employed kernels do not fully recover the underlying metric tensor field, they still define useful distance functions that account for the underlying nonlinear structure in the parameter space. Typically, the kernels used have been developed to operate on probability distributions or measures, for example, kernels based on (information-theoretic) divergence functions between two distributions (e.g., KL divergence [13]). In particular, Chan and Vasconcelos [14,15] used KL kernels on vector autoregressive (VAR) models to classify dynamic textures in video sequence analysis. Jebara *et al.* [16] proposed the probability product kernel (PPK), which can be seen as a dot product in the function space of two probability distributions. Bhattacharyya kernels, a special case of PPK, are related to the Hellinger distance between two functions. In Jebara *et al.* [16] PPK kernels were used to classify both LDS and HMM. Computation of KL and PPK kernels is analytically tractable only for simple classes of dynamical systems, such LDS and HMM. In general, their computation could be very expensive, since it can involve an infinite-dimensional integral over all possible state trajectories; Binet-Cauchy kernels could be seen as a counterpart of PPK kernel for deterministic dynamical systems [17,18]. In contrast to PPK, Binet-Cauchy kernels are defined as a dot product in the trajectory space. For deterministic systems, their trajectories are completely determined by their model parameters and the initial states. Extension of Binet-Cauchy kernels to stochastic dynamical systems is related to kernel mean embedding (KME) [19]. In KME, the embedding maps each distribution onto the Hilbert space induced by a chosen kernel [20]. In essence, PPK and KME are two kernels developed for classification of probability densities, distributions, or measures. When PPK and KME kernels are applied to classification of stochastic dynamical systems, they effectively operate on probability measures over system trajectories. Each measure is specified by a model parameter vector and a point estimate of this parameter vector is usually inferred from data so as to specify the corresponding measure.

Finally, we mention two kernels used in the literature for model-based time series classification that fall outside the LiMS framework since no individual models are inferred from individual time series. The Fisher kernel proposed by Jaakkola and Haussler [21] uses a single fixed time series model. Each time series is then represented by a tangent vector in the tangent space of that model. The AR kernel proposed by Cuturi and Doucet [22] is a marginalization kernel applied to AR models [23]. Each time series is represented by a (infinite-dimensional) "profile vector"—the AR likelihood for a set of model parameters, given that time series. The kernel between two time series is the dot product of the two corresponding profile functions, weighted by a prior distribution over the AR parameters.

In this paper, we present a general framework for classifying partially observed dynamical systems based on LiMS. One key ingredient of this framework is that given a model structure of parametrized dynamical systems, we represent each partially observed dynamical system (i.e., each time series) by a posterior distribution over model parameters. In contrast to all model-based approaches surveyed above, our approach takes into account the model uncertainty around each individual model. This is of particular relevance for the sparsely sampled time series as it could give rise to a considerable amount of uncertainty around the inferred model. For the classification task, we not only employ PPK and KME, two well-established distributional kernels, to classify those posterior distributions but also develop a distributional classifier specifically based on LiMS. The differences between inclusion of various types of treating model uncertainties (KME, PPK, LiMS) in the final stage of the classification process are studied and spelled out explicitly in Sec. IV.

To precisely present our LiMS framework, we clearly distinguish between model structure, structural model, model (instance), and (model) class, and model parameter. As dynamical systems are of our primary interest, the term "model structure" occurring in this paper refers to a particular causal relationship between state variables and their time derivative [see Eq. (1)]. This relationship is usually parametrized, which induces a metric parameter space. Structural model refers to a family of dynamical systems with a common model structure—parameter settings can vary. A dynamical system with a particular parameter setting is referred to as a model. A class of models is a subset of all models within a given structural model—their model parameters either occupy a domain in the parameter space or are sampled from a class-conditional probability distribution over the model parameters.

In the experiments we evaluate the performance of the three classifiers (LiMS, PPK, and KME) on two test beds, one representant of ODE (GnRH, Sec. V A), the other of SDE (SDW, Sec. V B). For a fair comparison all three

classifiers were implemented in the framework of kernel logistic regression (KLR, Section III B). Our study addressed two important issues for classifying partially observed dynamical systems (PODS):

(1) *The influence of model uncertainty on classification in the model space.* Model uncertainty arises when the underlying system is not completely observed. It is represented through posterior distribution over the model parameters inferred from the partial observations. It is natural to expect that the posterior over possible model parameters, given the observations, is a better (model space) representation of the observed time series than a single parameter, e.g., MAP point estimate. It is also natural to expect that the classification performance will increase with reducing model uncertainty. We compare the LiMS, PPK, and KME classifiers in terms of capability to deal with increased levels of model uncertainty quantified through entropy of the posterior distributions. We also use the level of observation noise σ or the number of observations n as surrogate uncertainty measures.

(2) *Performance degradation when the structural model used to represent the observed time series through posterior distributions over its model parameters is a reduced structural model of the true one generating the training and test data.* There can be several reasons for the inferential model structure to be different from the underlying data generating one. For example, in real-world applications, it is inevitable that there is a gap between the real world and the mathematical model developed to account for it. Alternatively, while the given mathematical model can be considered adequate, it is too complex and computationally expensive to simulate. To circumvent this problem, a reduced model structure could be used to represent time series, as long as it captures characteristics relevant for the given classification task. We compare the classification performance between different inferential model structures ranging from the full, multiple-compartment pathway ODE model to the trivial single compartment model. Analogous experiments were performed in the SDE case—SDW models representing data generated by stochastic multiwell systems.

The rest of this paper is organized as follows. We first formulate our framework in Sec. II. Section III presents an implementation of this framework by means of KLR. In Sec. IV we further establish connections between LiMS, PPK, and KME classifiers. Section V introduces two classes of dynamical systems used to validate our framework, and the experiments are detailed in Sec. VI. Finally, Sec. VII summarizes key research findings.

II. FRAMEWORK

A. Problem settings

First, a classification task is formulated as follows: Suppose we have N examples in the form of N labeled univariate or multivariate time series, denoted by $\{(\mathcal{Y}^k, c^k) : k = 1, \ldots, N\}$ where $\mathcal{Y}^k$ denotes the kth time series and c^k represents its binary label. As we do not assume that all time series are collected on a fixed, regular time grid, each time series $\mathcal{Y}^k$ is accompanied with a sequence of observation times $\{t_i^k\}_{i=1}^{L_k}$ at which the observations $\{\mathbf{y}_i^k\}_{i=1}^{L_k}$ are collected. Hence the kth time series is jointly represented by $\mathcal{Y}^k = (\mathbf{t}^k, \mathbf{Y}^k)$ with $\mathbf{t}^k = \{t_i^k : i = 1, \ldots, L^k\}$ and $\mathbf{Y}^k = \{\mathbf{y}_i^k : i = 1, \ldots, L^k\}$. Note that the length of time series L_k can vary across examples. However, the dimensionality d of the observed time series is assumed to be fixed. The task is to predict a label for a new time series $\mathcal{Y}$ of length L. Due to variability of observation times and length of the training time series, direct application of a vector-based classifier would not be suitable. Note that if the training time series were long enough and "suitably" sampled, one could represent each time series through, e.g., a vector of Fourier or wavelet coefficients. However, we do not wish to impose any such restrictions and in particular, we are interested in cases of short, sparsely, and irregularly sampled time series.

We propose to represent time series by a set of individual time series models from a given model class. In particular, since the observed time series can be noisy, short, and irregularly sampled, each time series will be represented as the posterior distribution over the models, given the time series itself and model prior.

B. Model-based representation

In our work, a dynamical system approach is adopted to model time series. In other words, we consider a given time series as a (possibly partial) observation of some underlying dynamical system from a parametric class of dynamical systems. In the following, we first introduce a mathematical representation of the class of dynamical systems considered in this work. Next, a model accounting for partial observations is formulated. Following this, we introduce a Bayesian approach for representing partially observed dynamical systems.

A continuous-time deterministic dynamical system can be mathematically represented as a multivariate ordinary differential equation (ODE):

$$\frac{d\mathbf{x}_t}{dt} = \mathbf{f}(\mathbf{x}_t; \boldsymbol{\psi}), \tag{1}$$

where $\mathbf{x}_t \in X \subset \mathbb{R}^D$ denotes D-dimensional state vector at time t. The mapping $\mathbf{f}$ specifies the dynamics of this system by defining the functional relation between state $\mathbf{x}_t$ and drift $\frac{d\mathbf{x}}{dt}$ at time t. This mapping is parametrized by $\boldsymbol{\psi}$. Note that model parameter $\boldsymbol{\psi}$ includes the initial state $\mathbf{x}_0$, unless $\mathbf{x}_0$ is assumed to be known.

A stochastic dynamical system can be considered as an ODE driven by a multivariate random process parametrized by covariance matrix $\boldsymbol{\Sigma}$. Each component of this process is a standard univariate Brownian motion scaled by square root of the corresponding diagonal term of $\boldsymbol{\Sigma}$. Its covariance structure at t is specified by the nondiagonal terms. It is equivalent to adding Gaussian noise to the drift. Mathematically, this system can be represented by a multivariate stochastic differential equation (SDE):

$$d\mathbf{x}_t = \mathbf{f}(\mathbf{x}_t; \boldsymbol{\psi})\, dt + \boldsymbol{\Sigma}^{1/2}\, d\mathbf{b}_t \tag{2}$$

where the vector $\mathbf{b}_t$ collects the D independent standard Brownian motions. A SDE's initial condition is specified by a probability distribution over $\mathbf{x}_0$, which is often assumed to be a Gaussian distribution with mean $\boldsymbol{\mu}_0$ and covariance matrix $\boldsymbol{\Sigma}_0$. As in ODEs, the initial condition specification is part of the model parameters $\boldsymbol{\psi}$.

All model parameters are collected in vector $\boldsymbol{\theta}$, i.e., for ODEs $\boldsymbol{\theta} = \boldsymbol{\psi}$ and for SDEs $\boldsymbol{\theta} = (\boldsymbol{\psi}, \boldsymbol{\Sigma})$.

Observations $\{\mathbf{y}_1, \mathbf{y}_2, \ldots\}$, $\mathbf{y}_t \in Y \subset \mathbb{R}^d$ from the underlying dynamical system's trajectory $\mathbf{x}_t$ are obtained through a measurement function $\mathbf{h}$:

$$\mathbf{y}_i = \mathbf{h}(\mathbf{x}_{t_i}) + \boldsymbol{\epsilon}_{t_i} \quad \text{for} \quad i = 1,2,\ldots, \tag{3}$$

where $\boldsymbol{\epsilon}_{t_i}$ denotes observation noise at time t_i. In general, $\mathbf{h}$ can be a parametric function with unknown parameters. Frequently, $\mathbf{h}$ represents a set of indicator functions which specify a subset of state variables that are directly observed. For clarity in formulating our general framework, we assume $\mathbf{h}$ to be an identity function. Observation noise $\boldsymbol{\epsilon}_t$ is often assumed to be independent and identically distributed (i.i.d.) Gaussian noise with zero mean and error covariance matrix $\mathbf{R}$. In this work, we treat model parameters and the parameters of observation noises differently. We assume that $\mathbf{R}$ is known—it can be determined from prior knowledge. For example, it is feasible to have a reliable estimate of measurement noise for experiments in physics. Of course, the inferential framework can be extended with an estimation or marginalization of the noise parameter $\mathbf{R}$, should such a need arise.

In the *learning in the model space* (LiMS) framework, the observed time series are represented through models parametrized via $\boldsymbol{\theta}$. Given a time series $\mathcal{Y} = (t_i, \mathbf{y}_i)_{i=1}^L$, a maximum likelihood (ML) estimate of $\boldsymbol{\theta}$ can be obtained by maximizing the likelihood function

$$p(\mathbf{Y}|\boldsymbol{\theta}, \mathbf{t}; \mathbf{R}) = \prod_{i=1}^{L} \mathcal{N}(\mathbf{y}_i|\mathbf{x}_t(\boldsymbol{\theta}), t_i, \mathbf{R}) \tag{4}$$

for an ODE system and

$$p(\mathbf{Y}|\boldsymbol{\theta}, \mathbf{t}, \mathbf{R}) = \mathbb{E}_{\mathbf{x}_t|\boldsymbol{\theta}}\left[\prod_{i=1}^{L} \mathcal{N}(\mathbf{y}_i|\mathbf{x}_t, t_i, \mathbf{R})\right] \tag{5}$$

for an SDE system. $\mathbb{E}_{\mathbf{x}_t|\boldsymbol{\theta}}[\ ]$ denotes expectation with respect to the probability measure over all possible system trajectories $\mathbf{x}_t$ specified by the SDE with model parameter vector $\boldsymbol{\theta}$. As trajectories of a continuous-time dynamical system are infinite dimensional, numerical integration over such trajectories is computationally very expensive and approximation is required for computing the expectation. However, this approach ignores uncertainty around the model estimate. In cases where only noisy and/or sparse data are available, any point estimate of the model parameter is not a sufficient representation of the partially observed dynamical system. Instead, we propose a Bayesian approach to represent model uncertainty via the posterior distribution of $\boldsymbol{\theta}$ as follows:

$$p(\boldsymbol{\theta}|\mathcal{Y}, \mathbf{R}) = p(\boldsymbol{\theta}|\mathbf{Y}, \mathbf{t}, \mathbf{R}) \propto p(\mathbf{Y}|\boldsymbol{\theta}, \mathbf{t}, \mathbf{R}) p(\boldsymbol{\theta}), \tag{6}$$

where $p(\boldsymbol{\theta})$ is the prior over $\boldsymbol{\theta}$. In most cases, computation of the normalizing term is analytically not tractable and the posterior has to be approximated by using a finite grid in the parameter space or by sampling or variational methods [24–27].

C. Classification framework

Formulation of a classifier for partially observed dynamical systems in the LiMS framework depends on the way the underlying systems are represented. We consider two different options:

(1) Representation through data $\mathcal{Y} = (\mathbf{Y}, \mathbf{t})$. The resulting classifier operates directly and solely on the data. A probabilistic classifier of this kind is formulated by defining the conditional probability $P(c|\mathcal{Y})$ that is used to predict label c in a probabilistic manner. Note that this classifier completely ignores the underlying model and thus is of disadvantage if the underlying model structure is known.

(2) Representation via posterior distributions over model parameters, $\pi(\boldsymbol{\theta}) \stackrel{\text{def}}{=} p(\boldsymbol{\theta}|\mathcal{Y}, \mathbf{R})$. Thus, the counterpart of $P(c|\mathcal{Y})$ is $P(c|\pi)$. The resulting classifier actually operates in the space of posterior distributions rather than in the model or data space. The posterior $\pi(\boldsymbol{\theta})$ is shaped by the metric structure in parameter space that encodes intrinsic information about the underlying dynamical system. In addition, it represents the uncertainty that arises due to a finite number of (possibly irregularly sampled) observations and observation noise.

Our classifier is of the latter type. In the following we derive the predictive distribution $P(c|\mathcal{Y})$ of class labels given the data and the model structure. Recall that given a model structure $\mathbf{f}$, data $\mathcal{Y}$ is assumed to be sampled from a hidden trajectory $\mathbf{x}_t$ generated by model $\mathbf{f}_{\boldsymbol{\theta}}$ (an unknown model parameter $\boldsymbol{\theta}$ sampled from a prior distribution over $\boldsymbol{\theta}$). Further, we assume that class label c is not conditional of $\mathbf{R}$, that is, $P(c|\mathcal{Y}) = P(c|\mathcal{Y}, \mathbf{R})$. To take this additional knowledge into account, we express $P(c|\mathcal{Y})$ as

$$P(c|\mathcal{Y}, \mathbf{R}) = \int \mathcal{D}_{\mathbf{x}} \int d\boldsymbol{\theta}\ P(c|\mathcal{Y}, \mathbf{x}_t, \boldsymbol{\theta})\ p(\mathbf{x}_t, \boldsymbol{\theta}|\mathcal{Y}, \mathbf{R}), \tag{7}$$

where the hidden trajectory $\mathbf{x}_t$ and unknown model parameter $\boldsymbol{\theta}$ are both marginalized out. The density $p(\mathbf{x}_t|\boldsymbol{\theta}, \mathcal{Y}; \mathbf{R})$ is defined with respect to the standard Brownian motion and $\int \mathcal{D}_{\mathbf{x}}$ represents the path integral over trajectories. The above formulation implies a classifier $P(c|\mathcal{Y}, \mathbf{x}_t, \boldsymbol{\theta})$ which utilizes the model instance $\boldsymbol{\theta}$, the trajectory $\mathbf{x}_t$ generated by $\mathbf{f}_{\boldsymbol{\theta}}$, and noisy observations $\mathcal{Y}$ assumed to be sampled from $\mathbf{x}_t$. Given $\mathbf{f}_{\boldsymbol{\theta}}$, $\mathbf{x}_t$ is either specified deterministically (in the case of ODEs), or is driven by a standard Brownian motion (in the case of SDEs). Assuming that no additional relevant information for the classification task could be extracted from observation noise or observation times (the noise and observation times processes are not conditional on the class label), all the relevant information in $(\mathcal{Y}, \mathbf{x}_t, \boldsymbol{\theta})$ for the class label prediction can be conditioned on the model parameter $\boldsymbol{\theta}$. Consequently, we replace $P(c|\mathcal{Y}, \mathbf{x}_t, \boldsymbol{\theta})$ with $P(c|\boldsymbol{\theta})$.

Equation (7) now reads

$$\begin{aligned} P(c|\mathcal{Y}, \mathbf{R}) &= \int d\mathbf{x}_t \int d\boldsymbol{\theta}\ P(c|\boldsymbol{\theta})\ \mathbf{p}(\mathbf{x}_t, \boldsymbol{\theta}|\mathcal{Y}, \mathbf{R}) \\ &= \int d\boldsymbol{\theta}\ P(c|\boldsymbol{\theta}) \int d\mathbf{x}_t\ p(\mathbf{x}_t, \boldsymbol{\theta}|\mathcal{Y}, \mathbf{R}) \\ &= \int d\boldsymbol{\theta}\ P(c|\boldsymbol{\theta})\, \pi(\boldsymbol{\theta}) \\ &= \mathbb{E}_{\pi(\boldsymbol{\theta})}[P(c|\boldsymbol{\theta})] \\ &= Q(c|\pi). \end{aligned} \tag{8}$$

Note that the classifier $Q(c|\pi)$ operates on posterior distributions π, but is based on the classifier $P(c|\boldsymbol{\theta})$ operating on

model parameters. In summary, the LiMS classifier defined by Eq. (8) is deduced from a conventional classifier operating on data under the assumption that class label c is not conditional of the observations and the unobserved trajectory.

In the following, we define the theoretical risk for $Q(c|\pi)$. Generally, theoretical risk for a classifier is defined through a joint distribution over the input and label spaces and a loss function quantifying the cost of misclassification. In our case, the joint distribution of (π,c) is written as $P(c)\mathcal{P}(\pi|c)$, where $\mathcal{P}$ denotes a distribution over distributions (random measure). The loss function we employ is the negative log likelihood, $-\ln P(c|\pi)$. The theoretical risk of $Q(c|\pi)$ can be written as

$$\mathcal{R}[Q(c|\pi)] = \mathbb{E}_{P(c)}[\mathbb{E}_{\mathcal{P}(\pi|c)}[-\ln Q(c|\pi)]]. \tag{9}$$

It is difficult to formulate $\mathcal{P}$ as a parametric generative model. For the classifier $Q(c|\pi)$, however, based on (8), we have $\mathcal{R}[Q(c|\pi)] = \mathcal{R}[P(c|\mathcal{Y})]$. The theoretical risk for $P(c|\mathcal{Y})$ is given by

$$\mathcal{R}[P(c|\mathcal{Y})] = \mathbb{E}_{P(c)}[\mathbb{E}_{p(\boldsymbol{\theta},\mathbf{x}_t,\mathbf{Y},\mathbf{R},\mathbf{t}|c)}[-\ln P(c|\mathcal{Y})]], \tag{10}$$

where

$$p(\boldsymbol{\theta},\mathbf{x}_t,\mathbf{Y},\mathbf{R},\mathbf{t}|c) = p(\boldsymbol{\theta}|c)p(\mathbf{t})p(\mathbf{R})p(\mathbf{X}_t|\boldsymbol{\theta})p(\mathbf{Y}|\mathbf{x}_t,\mathbf{t},\mathbf{R}). \tag{11}$$

A parametric formulation of the above theoretical risk is obtained by adopting (i) a parametric noise model for $p(\mathbf{Y}|\mathbf{x}_t,\mathbf{t},\mathbf{R})$; (ii) a parametric dynamical noise model $p(\mathbf{x}_t|\boldsymbol{\theta})$; (iii) a prior for the covariance of the observational noise $p(\mathbf{R})$; (iv) a point process for $p(\mathbf{t})$ in the observation window; and (v) an appropriate model for $p(\boldsymbol{\theta}|c)$. We write a density for $\mathbf{t}$ since it is defined with respect to the standard Poisson process.

III. IMPLEMENTATION

A. Computing the posterior distributions

For partially observed nonlinear dynamical systems the computation of posterior distributions is analytically not tractable. Therefore, the expectation over $\boldsymbol{\theta}$ with respect to $\pi(\boldsymbol{\theta})$ in Eq. (8) can only be computed via approximation. There exist two principled approximation strategies that have the required convergence properties: Approximation by sampling and finite-grid approximation. Note that when the number of samples or the size of finite grid go to infinity, the approximate posterior shall converge towards the exact one. Alternatively, there exist several variational procedures that approximate posterior densities under fixed-form assumption.

In the first approach (approximation by sampling), the posterior distribution is approximated by

$$\pi(\boldsymbol{\theta}) \approx \frac{1}{N_{\boldsymbol{\theta}}}\sum_{n=1}^{N_{\boldsymbol{\theta}}}\delta(\boldsymbol{\theta}-\boldsymbol{\theta}_n), \tag{12}$$

where $\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_{N_{\boldsymbol{\theta}}}$ are $N_{\boldsymbol{\theta}}$ parameter vectors which are independently sampled from $\pi(\boldsymbol{\theta})$. Accordingly, the classifier defined in Eq. (8) is approximated by

$$Q(c|\pi) \approx \frac{1}{N_{\boldsymbol{\theta}}}\sum_{n=1}^{N_{\boldsymbol{\theta}}}P(c|\boldsymbol{\theta}_n). \tag{13}$$

As the posterior distribution is only known up to normalizing constant, Markov chain Monte Carlo (MCMC) algorithms are the most efficient sampling method.

In the second approach (finite-grid approximation), one could first compute the unnormalized posterior density (that is, the product of normalized prior and likelihood densities) over a finite grid approximating the parameter space and then normalize those values into a multinomial distribution approximating the posterior density. We denote this grid and the multinomial posterior probabilities on the grid by

$$\mathcal{G}_{\boldsymbol{\theta}} = \left\{\boldsymbol{\theta}_1^{\mathcal{G}},\ldots,\boldsymbol{\theta}_{N_{\boldsymbol{\theta}}^{\mathcal{G}}}^{\mathcal{G}}\right\} \tag{14}$$

and

$$\left\{\pi^n = \frac{p\left(\mathcal{Y}|\boldsymbol{\theta}_n^{\mathcal{G}}\right)p\left(\theta_n^{\mathcal{G}}\right)}{\sum_{k=1}^{N_{\boldsymbol{\theta}}^{\mathcal{G}}}p\left(\mathcal{Y}|\boldsymbol{\theta}_k^{\mathcal{G}}\right)p\left(\theta_k^{\mathcal{G}}\right)} : n = 1,\ldots,N_{\boldsymbol{\theta}}^{\mathcal{G}}\right\}, \tag{15}$$

respectively. The resulting approximate classifier is given by

$$Q(c|\pi) \approx \sum_{n=1}^{N_{\boldsymbol{\theta}}^{\mathcal{G}}}\pi^n P\left(c|\boldsymbol{\theta}_n^{\mathcal{G}}\right). \tag{16}$$

For SDE, however, the marginal likelihood for each parameter vector on the grid is analytically not tractable and thus the likelihood is not normalized. To solve this problem at low computational cost, we employ the variational Gaussian process approximation method for computing the approximate marginal likelihood [26]. In the literature, variational Bayes (VB) methods have also been adopted for approximate inference in partially observed dynamical systems. Particularly, in the field of fMRI data analysis, a family of partially observed dynamical systems termed as dynamical causal model (DCM) has been developed for modeling and analysis of fMRI time series. Moreover, the VB approaches equipped with Laplace approximation were employed to infer DCM from fMRI data [28]. In our case we use a variational method not to approximate posterior over the models, but to approximate posterior over state trajectories in the case of stochastic systems.

B. Kernel logistic regression for binary classification

In the following, we first briefly introduce kernel logistic regression (KLR) as a (binary) classifier for vectors (e.g., model parameter $\boldsymbol{\theta}$). We then present an extension of KLR for distributions so that the classifier can be directly applied to posteriors $\pi(\boldsymbol{\theta})$.

A binary KLR classifier operating on $\boldsymbol{\theta}$s is defined via

$$P(c=1|\boldsymbol{\theta}) = \zeta(\mathbf{w}^{\mathsf{T}}\boldsymbol{\Phi}(\boldsymbol{\theta})), \tag{17}$$

where $\zeta(\,)$ denotes a sigmoid function [that is, for real number a, $\zeta(a)$ is defined as $\frac{1}{1+\exp(-a)}$], $\mathbf{w}$ is an m-dimensional classifier parameter, and $\boldsymbol{\Phi}$ represents a (nonlinear) mapping of the $D_{\boldsymbol{\theta}}$-dimensional model parameter vector $\boldsymbol{\theta}$ to an m-dimensional feature space:

$$\boldsymbol{\Phi} : \boldsymbol{\theta} \longmapsto \left[K\left(\boldsymbol{\theta},\boldsymbol{\theta}_1^{\mathcal{F}}\right),\ldots,K\left(\boldsymbol{\theta},\boldsymbol{\theta}_m^{\mathcal{F}}\right)\right]^{\mathsf{T}}, \tag{18}$$

where $K(\,,\boldsymbol{\theta}_1^{\mathcal{F}}),\ldots,K(\,,\boldsymbol{\theta}_m^{\mathcal{F}})$ act as basis functions in the parameter space centered at parameter vectors from

$\mathcal{F}_{\boldsymbol{\theta}} = \{\boldsymbol{\theta}_1^{\mathcal{F}}, \ldots, \boldsymbol{\theta}_m^{\mathcal{F}}\}$. In this work, we define $K(\ ,\ )$ by a Gaussian kernel,

$$K(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2) = \exp\left(-\frac{\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|_2^2}{\rho}\right), \tag{19}$$

where $\|\ \|_2$ is the Euclidean norm and $\rho > 0$ is a scale parameter. Note that in our case, kernels are simply used as basis functions for the nonlinear classifier construction. In other words, we do not require a Mercer kernel, as our model fitting does not involve a linear classifier construction in the feature space induced by the kernel.

For learning the classifier parameter $\mathbf{w}$, a training set of N labeled model parameters $S = \{(\boldsymbol{\theta}_1, c_1), \ldots, (\boldsymbol{\theta}_N, c_N)\}$, $c_i \in \{0,1\}$, would be used to obtain the maximum likelihood estimate (MLE) of $\mathbf{w}$:

$$\hat{\mathbf{w}}^{\mathbf{MLE}} = \arg\max_{\mathbf{w}} \prod_{k=1}^{N} z_k^{c_k}(1 - z_k)^{1-c_k} \quad \text{with}$$
$$z_k = \zeta(\mathbf{w}^{\mathsf{T}} \boldsymbol{\Phi}(\boldsymbol{\theta}_k)). \tag{20}$$

This is equivalent to minimizing the cross entropy error,

$$E_{\boldsymbol{\theta}}(\mathbf{w}|S) = -\sum_{k=1}^{N} \ln P(c_k|\boldsymbol{\theta}_k; \mathbf{w}) \tag{21}$$
$$= -\sum_{k=1}^{N} c_k \ln(z_k) + (1 - c_k)\ln(1 - z_k) \tag{22}$$
$$= \sum_{k=1}^{N}\left(-1 \sum_{l=0}^{1} Q_k^l \ln\left(P_k^l\right)\right) \tag{23}$$
$$= \sum_{k=1}^{N} H(Q_k, P_k) \tag{24}$$

with

$$Q_k^l = \begin{cases} c_k & \text{if } l = 0 \\ 1 - c_k & \text{if } l = 1 \end{cases} \quad \text{and}$$
$$P_k^l = \begin{cases} z_k & \text{if } l = 0 \\ 1 - z_k & \text{if } l = 1 \end{cases}, \tag{25}$$

where $H(Q_k, P_k)$ denotes the cross entropy of probability distributions q_k and p_k.

For a gradient-based minimization of $\mathbf{E}$ with respect to $\mathbf{w}$, the gradient is computed as

$$\nabla_{\mathbf{w}}\mathbf{E} = \sum_{k:c_k=1} (z_k - 1)\boldsymbol{\Phi}(\boldsymbol{\theta}_k) + \sum_{k:c_k=0} z_k \boldsymbol{\Phi}(\boldsymbol{\theta}_k). \tag{26}$$

Next, we present a gradient-based algorithm to learn a classifier that operates on the posterior distributions using a training set given as

$$V = \{(\pi_1(\boldsymbol{\theta}), c_1), \ldots, (\pi_N(\boldsymbol{\theta}), c_N)\}. \tag{27}$$

Recall that this classifier is defined as expectation of a probabilistic base classifier operating on the model parameter with respect to the posterior distribution [see Eq. (8)]. In this setting, the base classifier's free parameter automatically becomes the free parameter of the distributional classifier. On the other hand, the base classifier is trained by maximizing the likelihood function of class label assigned to a model parameter. Naturally, the classifier that classifies the posterior distribution should be trained by maximizing the expectation of the likelihood function with respect to the posterior distribution. Therefore, the classifier parameter is obtained by minimizing the cross entropy error,

$$E_{\pi}(\mathbf{w}|V) = -\sum_{k=1}^{N} \ln\left(\int d\boldsymbol{\theta}\ \pi_k(\boldsymbol{\theta}) P(c_k|\boldsymbol{\theta}; \mathbf{w})\right). \tag{28}$$

The approximate cross entropy error is computed by

$$\hat{E}_{\pi}(\mathbf{w}|V) = -\sum_{k=1}^{N} \ln\left(\sum_{n=1}^{N_{\boldsymbol{\theta}}^{\mathcal{G}}} \pi_k^n P\left(c_k\middle|\boldsymbol{\theta}_n^{\mathcal{G}}; \mathbf{w}\right)\right), \tag{29}$$

where π_k^n denotes the normalized posterior weight on the nth grid point for the kth posterior. The corresponding gradient is given by

$$\nabla_{\mathbf{w}}\hat{E}_{\pi} = \sum_{n=1}^{N_{\boldsymbol{\theta}}^{\mathcal{G}}} \left[Z_n^1\left((z_n - 1)\boldsymbol{\Phi}\left(\boldsymbol{\theta}_n^{\mathcal{G}}\right)\right) + Z_n^0\left(z_n \boldsymbol{\Phi}\left(\boldsymbol{\theta}_n^{\mathcal{G}}\right)\right)\right], \tag{30}$$

where $z_n = \zeta(\mathbf{w}^{\mathsf{T}} \boldsymbol{\Phi}(\boldsymbol{\theta}_n^{\mathcal{G}}))$,

$$Z_n^1 = \sum_{k:c_k=1} \frac{\pi_k^n z_n}{\sum_{l=1}^{N_{\boldsymbol{\theta}}^{\mathcal{G}}} \pi_k^l z_l} \quad \text{and} \quad Z_n^0 = \sum_{k:c_k=0} \frac{\pi_k^n (1 - z_n)}{\sum_{l=1}^{N_{\boldsymbol{\theta}}^{\mathcal{G}}} \pi_k^l (1 - z_l)}. \tag{31}$$

Note that $\boldsymbol{\Phi}(\boldsymbol{\theta}_n^{\mathcal{G}})$ is a m-dimensional vector whose jth component is given by $K(\boldsymbol{\theta}_n^{\mathcal{G}}, \boldsymbol{\theta}_j^{\mathcal{F}})$. The two grids on the parameter space, $\mathcal{G}$ (posterior support) and $\mathcal{F}$ (basis functions centers), can be the same, or $\mathcal{F}$ can be a subset of $\mathcal{G}$.

IV. CONNECTION TO RELATED WORKS

In literature, most distributional classifiers combine an existing kernel-based classifier, such as SVM, with a kernel that is defined on the space of distributions. An example of such a kernel is the so-called probability product kernel [16],

$$K_{\mathrm{PPK}}(\pi_1, \pi_2) = \int_{\Theta} d\boldsymbol{\theta}\ \pi_1^{\alpha}(\boldsymbol{\theta})\pi_2^{\alpha}(\boldsymbol{\theta}), \tag{32}$$

where π_1 and π_2 are two distributions over a metric space Θ and $\alpha > 0$ is a tempering parameter. In recent literature, another kernel on distributions has been introduced based on Hilbert space embedding [19,20]. Given a universal kernel $k : \Theta \times \Theta \longrightarrow \mathbb{R}$, there exists an injective mapping from distribution space Q to feature space,

$$\mu_Q : Q \to \mathcal{H}, \quad \pi \longmapsto \int_{\Theta} k(\boldsymbol{\theta},)\pi(\boldsymbol{\theta})\, d\boldsymbol{\theta}. \tag{33}$$

This mapping is called kernel mean embedding (KME). As the embedding is bijective, no information encoded in the probability distribution is lost through the mapping. The mapping in turn defines a kernel on probability distributions, $K : \mathcal{Q} \times \mathcal{Q} \longrightarrow \mathbb{R}$:

$$K_{\mathrm{KME}}(\pi_1, \pi_2) = \langle \mu_{\pi_1}, \mu_{\pi_2} \rangle_{\mathcal{H}}$$
$$= \int_{\boldsymbol{\theta}\in\Theta} d\boldsymbol{\theta} \int_{\boldsymbol{\eta}\in\Theta} d\boldsymbol{\eta}\ \ \pi_1(\boldsymbol{\theta})\pi_2(\boldsymbol{\eta})k(\boldsymbol{\theta}, \boldsymbol{\eta}). \tag{34}$$

We compare these two distributional classifiers (one based on K_{PPK}, the other one based on K_{KME}) with our classifier in terms of their predictive class distributions, given a test input (distribution) π:

(i) probabilistic classifier based on probability product kernel (PPK):

$$P(c=1|\pi) = \zeta\left(\int_{\Theta} \pi^{\alpha}(\boldsymbol{\theta})\left[\underbrace{\sum_{i=1}^{N} v_i \pi_i^{\alpha}(\boldsymbol{\theta})}_{\Upsilon_{\mathrm{PPK}}(\boldsymbol{\theta};\mathbf{v})}\right] d\boldsymbol{\theta}\right) \quad (35)$$

$$= \zeta(\mathbb{E}_{\pi}[\Upsilon_{\mathrm{PPK}}(\boldsymbol{\theta};\mathbf{v})]), \quad (36)$$

where $\Upsilon_{\mathrm{PPK}}(\boldsymbol{\theta};\mathbf{v})$ denotes the function to be learned (by adjusting the free parameter $\mathbf{v}$) for classifying distributions;

(ii) probabilistic classifier based on kernel mean embedding (KME):

$$\begin{aligned} P(c=1|\pi) &= \zeta\left(\int_{\boldsymbol{\theta}\in\Theta} \pi(\boldsymbol{\theta})\left[\int_{\boldsymbol{\eta}\in\Theta}\left[\sum_{i=1}^{L} v_i \pi_i(\boldsymbol{\eta})\right] k(\boldsymbol{\theta},\boldsymbol{\eta})\, d\boldsymbol{\eta}\right] d\boldsymbol{\theta}\right) \\ &= \zeta\left(\int_{\boldsymbol{\theta}\in\Theta} \pi(\boldsymbol{\theta})\left[\sum_{i=1}^{L} v_i \int_{\boldsymbol{\eta}\in\Theta} \pi_i(\boldsymbol{\eta}) k(\boldsymbol{\theta},\boldsymbol{\eta})\, d\boldsymbol{\eta}\right] d\boldsymbol{\theta}\right) \quad (37) \\ &= \zeta\left(\int_{\boldsymbol{\theta}\in\Theta} \pi(\boldsymbol{\theta}) \underbrace{\left[\sum_{i=1}^{L} v_i \tilde{\pi}_i(\boldsymbol{\theta})\right]}_{\Upsilon_{\mathrm{KME}}(\boldsymbol{\theta};\mathbf{v})} d\boldsymbol{\theta}\right) \quad (38) \\ &= \zeta(\mathbb{E}_{\pi}[\Upsilon_{\mathrm{KME}}(\boldsymbol{\theta};\mathbf{v})]), \quad (39) \end{aligned}$$

where $\tilde{\pi}_i$ are kernel-smoothed posteriors π_i and $\Upsilon_{\mathrm{KME}}(\boldsymbol{\theta};\mathbf{v})$ is the function to be learned;

(iii) probabilistic classifier proposed in this work [Eq. (8)]:

$$P(c=1|\pi) = \int_{\Theta} \pi(\boldsymbol{\theta}) \zeta\Bigg(\underbrace{\sum_{i=1}^{m} w_i k\left(\boldsymbol{\theta},\boldsymbol{\theta}_i^{\mathcal{F}}\right)}_{\Upsilon_{\mathrm{LiMS}}(\boldsymbol{\theta};\mathbf{w})}\Bigg) d\boldsymbol{\theta} \quad (40)$$

$$= \mathbb{E}_{\pi}[\zeta[\Upsilon_{\mathrm{LiMS}}(\boldsymbol{\theta};\mathbf{w})]], \quad (41)$$

where $\Upsilon_{\mathrm{LiMS}}(\boldsymbol{\theta};\mathbf{w})$ is learned by adjusting the free parameter $\mathbf{w}$.

We implemented both PPK and KME within the KLR framework. These KLR classifiers are trained exactly in the same way as the KLR classifier operating on model parameter vectors. The only difference between them lies in which basis functions are used in the classifier construction (that is, Gaussian kernel, posterior distributions from the training set, or those posterior distributions smoothed by Gaussian kernel).

To see a deeper connection between the three classifiers above, consider first the usual setting of kernel logistic regression,

$$P(c=1|\boldsymbol{\theta}) = \zeta[\Upsilon(\boldsymbol{\theta})]. \quad (42)$$

This can be interpreted as follows: The model imposes a smooth field (natural parameter of Bernoulli distribution) $\Upsilon(\boldsymbol{\theta})$ over the inputs $\boldsymbol{\theta}$.

The field assigns to each input a real number that expresses the "strength" with which that particular input wants to belong to class $+1$. Pushing the field through the link function ζ creates a new field $\zeta[\Upsilon(\boldsymbol{\theta})]$ over the inputs, assigning to each $\boldsymbol{\theta}$ the probability with which it belongs to class $+1$.

In case our inputs are not individual models $\boldsymbol{\theta}$, but (posterior) distributions π over the models, the classifier (42) can be generalized in two ways:

(1) Use the posterior distribution π to average over individual natural parameters $\Upsilon(\boldsymbol{\theta})$ to create the overall mean natural parameter $\mathbb{E}_{\pi}[\Upsilon(\boldsymbol{\theta})]$. This can then be passed through the link function ζ to calculate the class $+1$ probability for π, $\zeta(\mathbb{E}_{\pi}[\Upsilon(\boldsymbol{\theta})])$. This scenario can be described as forming an (infinite) ensemble to form the overall opinion about the strength of π belonging to class $+1$ and only then turning it into the class probability. This option is taken by the classifiers based on probability product kernel and kernel mean embedding, (36) and (39), respectively.

(2) Use the posterior distribution π to average over individual class probabilities $\zeta[\Upsilon(\boldsymbol{\theta})]$ to form the overall class probability $\mathbb{E}_{\pi}(\zeta[\Upsilon(\boldsymbol{\theta})])$ of π. This corresponds to creating an ensemble of probabilistic classifiers $\zeta[\Upsilon(\boldsymbol{\theta})]$ acting on individual models $\boldsymbol{\theta}$, as done by the proposed classifier [see (41)].

One can view the latter approach $\mathbb{E}_{\pi}(\zeta[\Upsilon(\boldsymbol{\theta})])$ as a regularization of the former one $\zeta(\mathbb{E}_{\pi}[\Upsilon(\boldsymbol{\theta})])$. Loosely speaking, when collecting ensemble votes to form an opinion about the probability of class $+1$ given π, $\mathbb{E}_{\pi}(\zeta[\Upsilon(\boldsymbol{\theta})])$ ignores the (potentially huge) differences between individual natural parameters $\Upsilon(\boldsymbol{\theta})$ giving negligible differences in the probabilities $\zeta[\Upsilon(\boldsymbol{\theta})]$ because of the saturation regions at both extremes of the link function ζ. This effectively collapses input regions of models $\boldsymbol{\theta}$ with high positive field values into a single high class probability region. Analogously, regions of models $\boldsymbol{\theta}$ with low negative values will be identified into a low class probability region.

Another point of view is to compare the models for the field $\Upsilon(\boldsymbol{\theta})$ utilized in the three classifiers. In all cases the fields are modelled as linear combinations of basis functions. Because kernels of the classifiers based on probability product kernel and kernel mean embedding operate on full distributions, the basis functions for modeling the field $\Upsilon(\boldsymbol{\theta})$ are the (possibly tempered) training posterior distributions π_i^{α} or their kernel-smoothed versions $\tilde{\pi}_i$, respectively [see (35) and (38)]. In contrast, the proposed classifier (41) models the field $\Upsilon(\boldsymbol{\theta})$ in a less constrained framework of kernel regression as a linear combination of kernel basis functions $k(\ ,\boldsymbol{\theta}_i^{\mathcal{F}})$ [see (40)]. In particular, no assumption is made that the field should lie in the span of the training distributions π_i^{α} or their smoothed versions $\tilde{\pi}_i$.

V. TEST BEDS

In this work, we validate our general framework using two example dynamical systems: Gonadotropin-Releasing Hormone Signalling model (GnRH) [29] and stochastic double-well systems (SDW) [30]. GnRH is an example of ordinary differential equation (ODE) systems and SDW is an example of stochastic differential equation (SDE). GnRH is also an example of the biological pathway or compartment model.

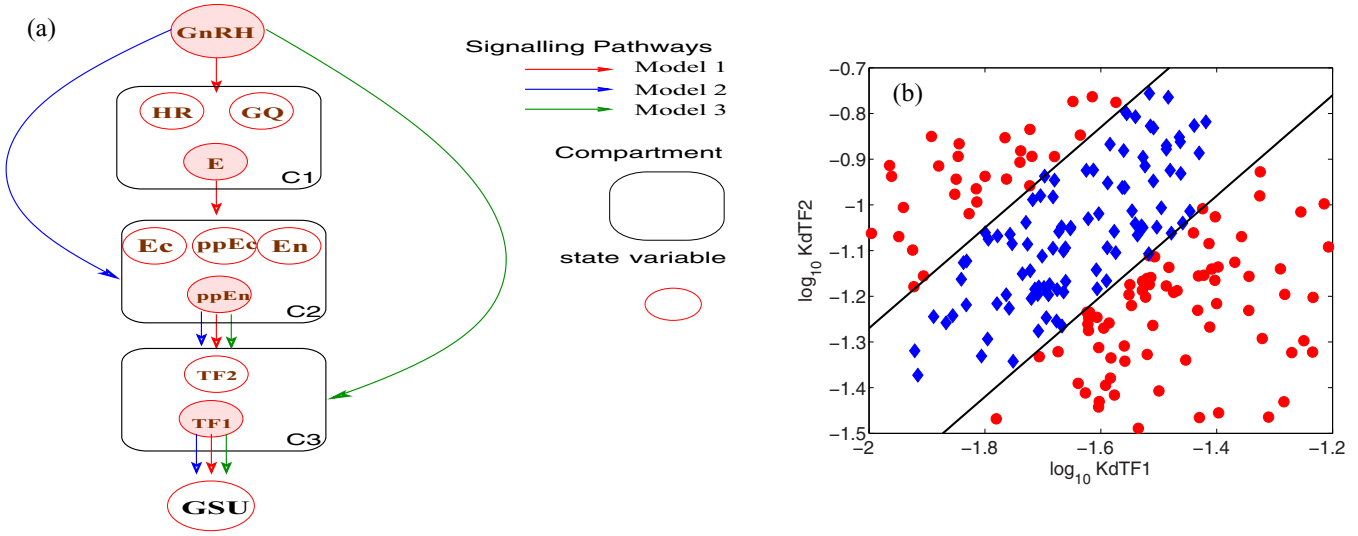

FIG. 1. (a) Schematical representation of three nested GnRH signaling model structures ($M1$, $M2$, and $M3$). The signaling pathway in these model structures is highlighted by the flow of red, blue, and green arrows, respectively. Each model structure is comprised of GnRH signal as the driving input, GSU time series as the measurable output, and one to three compartments along its signaling pathway. (b) Two classes of GnRH model parameter pairs (K_{dTF_1}, K_{dTF_2}) that determine subject-specific functional relationship between GnRH pulse frequency and the integrated GSU measure: (1) class of normal subjects with bell-shaped frequency-response relationship (blue diamonds) and (2) class of abnormal subjects with simple frequency dependency of the GSU measure (red disks). These two classes are separated by two straight lines in the log-log parameter space.

A. GnRH signaling model

Mathematically, GnRH signaling model is represented by an ODE system with 11 state variables. These state variables include concentrations of gonadotropin releasing hormones ([GnRH]) and gonadotropin hormones ([GSU]) as the driving input and measurable output of this structural model (respectively). The remaining state variables can be grouped into three compartments along the signaling pathway: (1) C1 for GnRH binding process; (2) C2 for extracellular signal regulated kinase (ERK) activation; and (3) C3 for transcription factor (TF) activation. We refer to this model structure as $M1$ and further introduce two reduced model structures derived from $M1$ in the following. In Fig. 1, we highlight the signaling pathway in $M1$ by red arrows. By removing the compartment C2 from the pathway, we obtain a two-compartment model structure denoted by $M2$ (see blue arrows in Fig. 1). When we further remove C3 from the pathway, $M2$ is reduced to $M3$ (see green arrows in Fig. 1) in which GnRH signals directly modulate stimulation of transcriptional activation. Actually, $M1$, $M2$, and $M3$ make up a hierarchy of three nested GnRH model structures. Their corresponding structural models can be used as building blocks to represent subject-specific behavior of [GSU] response to varying [GnRH] frequencies at different complexity levels.

GnRH signal is the chemical signal which stimulates the reproductive endocrine system. This signal is modeled by

$$\frac{d[\mathrm{GnRH}]}{dt} = -[\mathrm{GnRH}] + p_{\mathrm{GnRH}}\{H(t \bmod f^{-1}) - H[(t \bmod f^{-1}) - t_p]\}, \tag{43}$$

where p_{GnRH} is the GnRH pulse magnitude, f is the pulse frequency, and t_p is the pulse duration. In this work, we set p_{GnRH} to be a constant (i.e., $p_{\mathrm{GnRH}} = 0.1$) and treat both f and t_p as model parameters. Equation (43) can also be formulated as a mapping between time t and $[\mathrm{GnRH}]_t$ by

$$[\mathrm{GnRH}]_t = \mathbf{g}(t; f, t_p). \tag{44}$$

The amount of TF_1 and TF_2, denoted by $[\mathrm{TF}_1]$ and $[\mathrm{TF}_2]$, are two state variables in C3 which modulate the dynamics of the GSU expression as follows:

$$\frac{d[\mathrm{GSU}]}{dt} = K_{\mathrm{complex}}\left(\frac{\frac{[\mathrm{TF}_1]}{K_{d_{\mathrm{TF}_1}}}\frac{[\mathrm{TF}_2]}{K_{d_{\mathrm{TF}_2}}}[DNA_{\mathrm{TOT}}]^2}{\left(1 + \frac{[\mathrm{TF}_1]}{K_{d_{\mathrm{TF}_1}}} + \frac{[\mathrm{TF}_2]}{K_{d_{\mathrm{TF}_2}}}\right)^2}\right) - d_{[\mathrm{GSU}]}[\mathrm{GSU}], \tag{45}$$

where $K_{d_{\mathrm{TF}_1}}$ and $K_{d_{\mathrm{TF}_2}}$ are the dissociation constants of $[\mathrm{TF}_1]$ and $[\mathrm{TF}_2]$, respectively. They are both considered as model parameters. The remaining model parameters are set values reported in the literature [29]. In summary, the GnRH signaling model structure has one observable and four free model parameters. The observable is GSU and the model parameters are as follows: GnRH pulse frequency f, GnRH pulse duration t_p, the dissociation constant of $[\mathrm{TF}_1]$, $K_{d_{\mathrm{TF}_1}}$, and the dissociation constant of $[\mathrm{TF}_2]$, $K_{d_{\mathrm{TF}_2}}$. Also, the GnRH model can be formulated as a mapping between $[\mathrm{GnRH}]_t$ and $[\mathrm{GSU}]_t$ by

$$[\mathrm{GSU}]_t = \mathbf{M}\big([\mathrm{GnRH}]_t; K_{d_{\mathrm{TF}_1}}, K_{d_{\mathrm{TF}_2}}\big) = \mathbf{M}\big[\mathbf{g}(t; f, t_p); K_{d_{\mathrm{TF}_1}}, K_{d_{\mathrm{TF}_2}}\big], \tag{46}$$

where the mapping $\mathbf{M}$ can be set as one of the three nested GnRH model structures (that is $M1$, $M2$, or $M3$).

It is widely accepted that the reproductive system is controlled via GnRH pulse frequency. This frequency varies

under different physiological conditions, affecting the transcription of GSU and secretion of reproductive hormones that are crucial for the physiology of the reproductive system. GnRH frequency decoding mechanisms vary under normal and pathological conditions, but two main possibilities exist: (1) increasing pulse frequency simply increases output (GSU) until a maximal response is maintained with continuous stimulation [see Fig. 6(a) in Trapeva-Atanasova *et al.* [29]]; and (2) pulsatile stimuli may elicit maximal responses at submaximal frequencies, generating a bell-shaped frequency-response relationship [see Fig. 6(b) in Trapeva-Atanasova *et al.* [29]]. In this work, we utilize these two mechanisms to define two classes of subjects: "abnormal" [mechanism (1)] and "normal" [mechanism (2)] subjects. It has also been shown in Trapeva-Atanasova *et al.* [29] that the frequency-response behavior of GnRH models is determined by $K_{d_{\mathrm{TF}_1}}$ and $K_{d_{\mathrm{TF}_2}}$, but not by t_p. Fig. 1(b) shows that in the space of ($\log K_{d_{\mathrm{TF}_1}}$, $\log K_{d_{\mathrm{TF}_2}}$), there exist three linearly separated domains in which only one of two frequency-response behaviors (linear or bell shaped) is observed. The domain in the middle represents the normal subjects, whereas both remaining domains represent the abnormal subjects. It should be noted that those four parameters of the GnRH model actually play different roles in the classification task. The dissociation constants $K_{d_{\mathrm{TF}_1}}$ and $K_{d_{\mathrm{TF}_2}}$ are two class-defining model parameters and their values vary under different medical conditions. Therefore, it is of most importance to account for uncertainty around these two parameters via posterior distribution. In contrast, GnRH pulse frequency f is used to generate [GSU] time series under different physiological conditions. Those [GSU] time series generated under different f values are needed to accurately infer $K_{d_{\mathrm{TF}_1}}$ and $K_{d_{\mathrm{TF}_2}}$. Note that in the classification task, f is a controlling parameter and their values are known. The GnRH pulse duration t_p accounts for the individual variability that is related neither to the medication conditions nor to the physiological conditions. Therefore, it is not conditioned on class labels and it also remains a constant with varying f for each subject.

As subjects from two different classes differ in how their [GSU] responses change with varying pulse frequency, it is thus not sufficient to represent individual subjects by Eq. (46). Instead, we define the following composite GnRH model to represent those subjects:

$$\begin{aligned}[\mathrm{GSU}]_t^1 &= \mathbf{M}\big(\mathbf{g}(t; f_1, t_p); K_{d_{\mathrm{TF}_1}}, K_{d_{\mathrm{TF}_2}}\big),\\ &\vdots\\ [\mathrm{GSU}]_t^6 &= \mathbf{M}\big(\mathbf{g}(t; f_6, t_p); K_{d_{\mathrm{TF}_1}}, K_{d_{\mathrm{TF}_2}}\big),\end{aligned} \tag{47}$$

where six different pulse frequencies in this model are set as follows: $f_1 = \frac{1}{8}$, $f_2 = \frac{1}{4}$, $f_3 = \frac{1}{2}$, $f_4 = 1$, $f_5 = 2$, and $f_6 = 4$. The chosen frequencies adequately cover the entire permissible range. Note that the measurable output of this composite GnRH model structure is $([\mathrm{GSU}]_t^1, \ldots, [\mathrm{GSU}]_t^6)^\intercal$ and $(K_{dTF_1}, K_{dTF_2}, t_p)$ is its model parameter triple.

For the classification task, we use the composite model structure given by Eq. (47) to define two classes of subject-specific GnRH models and generate training and testing data accordingly. When Eq. (47) is used as a data-generating structural model, we set $\mathbf{M}$ by $M1$ because the full model structure is considered as the most realistic mathematical model among $M1$, $M2$, and $M3$. The model parameters K_{dTF_1}, K_{dTF_2}, and t_p are sampled from three separate truncated Gaussian distributions. For details, we refer to Table I in Sec. VI B. Based on their K_{dTF_1} and K_{dTF_2} values, all subjects are labeled as "healthy" or "abnormal" as illustrated in Fig. 1.

TABLE I. The truncated Gaussian distributions of three GnRH model parameters (i.e., $\log K_{d_{\mathrm{TF}_1}}$, $\log K_{d_{\mathrm{TF}_2}}$, and t_p) used for generating the training and testing set of GnRH models.

Parameter	Mean	Variance	Lower bound	Upper bound
$\log K_{d_{\mathrm{TF}_1}}$	−1.6	0.2	−2.0	0.2
$\log K_{d_{\mathrm{TF}_2}}$	−1.1	0.2	−1.5	0.2
t_p	7.5	0.8333	5	10

However, for the inferential task inside the classification process, we set $\mathbf{M}$ to $M1$, $M2$, or $M3$ as three inferential composite GnRH models used in the validation experiment. This allows us to assess whether the gap between the data-generating and inferential model structure could hamper the classification performance of our LiMS framework.

B. Stochastic double-well systems

The stochastic double-well (SDW) system is mathematically defined as

$$dx_t = \underbrace{4(x_t - a)\big(d^2 - x_t^2\big)}_{f(x_t)} + \kappa^2\, db_t, \tag{48}$$

where b_t represents the univariate standard Brownian motion and $\boldsymbol{\theta} = (d, \kappa, a)$ collects the three model parameters, namely the well location parameter d, well asymmetry parameter a, and standard deviation κ of the dynamical noise. Equation (48) shows that the drift term $f(x_t)$ is not explicitly time dependent. Therefore, the underlying dynamics is governed by the potential $u(x)$ with $f(x) = -\nabla_x u(x)$. Moreover, the equilibrium probability distribution of its state x is given by $p^{\mathbf{eq}}(x) \propto \exp(-\frac{u(x)}{\kappa^2})$ [31]. The potential corresponding to Eq. (48) is given by

$$u(x) = x^4 - \tfrac{4}{3}ax^3 - 2d^2x^2 + 4ad^2x. \tag{49}$$

The equilibrium probability distribution of two example SDWs is shown in Fig. 2(a). We can see that there exist two metastable states located at $x = d$ and $x = -d$. The larger is the dynamical noise variance κ^2, the more frequent are the transitions from one metastable state to the other. Figure 2 shows that the peak probability for $\kappa = 1.0$ (red solid curve) is larger than that for $\kappa = 1.5$ (blue dash-dot curve). For positive well asymmetry parameter a, the transition from $x = -d$ to $x = d$ is more likely than the transition in the opposite direction. As a result, the equilibrium probability at $x = d$ is higher than that at $x = -d$ (see red solid curve in Fig. 2). Analogously, the equilibrium probability at $x = -d$ is higher than that at $x = d$ for negative well asymmetry parameter (see blue dash-dot curve in Fig. 2). The dynamics of double-well systems is dominated by switching between the two wells. We also study more complex multiwell systems where the potential has more than two wells. An example of such a

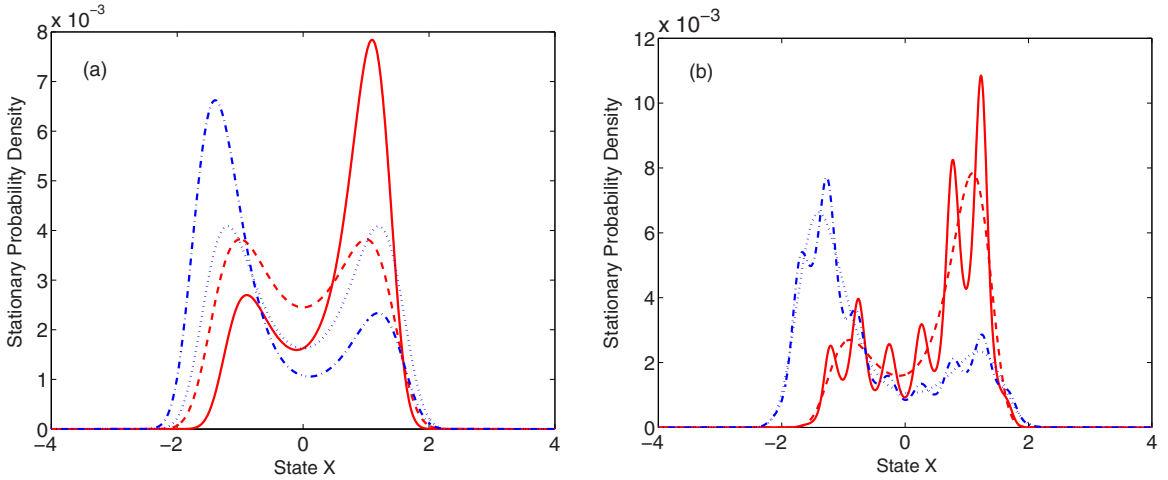

FIG. 2. (a) Equilibrium probability distribution of states x for four example stochastic double-well systems with $(d,\kappa,a) = (1.0,1.0,0.1)$ (red solid curve), $(d,\kappa,a) = (1.3,1.5,-0.1)$ (blue dash-dot curve), $(d,\kappa,a) = (1.0,1.5,0)$ (red dashed curve), and $(d,\kappa,a) = (1.2,1.5,0)$ (blue dotted curve). (b) The same as in (a) but for stochastic multiwell systems.

multiwell system dominated by an overall two-well structure [wells in positive range of x are generally deeper than those in the negative range (or vice versa)] is given below [see also Fig. 2(b)]:

$$dx_t = -\nabla_x \tilde{u}(x) + \kappa\, dw \quad \text{with}$$
$$\tilde{u}(x) = u(x) + \tfrac{1}{2}\cos(4\pi x), \tag{50}$$

where $\tilde{u}$ denotes the perturbed potential.

In this work, we formed two classes of SDWs through two class-conditional Gaussian distributions in the parameter space as follows: $(\bar{d}_1 + \epsilon_d, \bar{\kappa}_1 + \epsilon_\kappa, \bar{a}_1)$ for class 1 and $(\bar{d}_0 + \epsilon_d, \bar{\kappa}_0 + \epsilon_\kappa, \bar{a}_0)$ for class 0, where $(\bar{d}_1, \bar{\kappa}_1, \bar{a}_1)$ and $(\bar{d}_0, \bar{\kappa}_0, \bar{a}_0)$ denote the class-conditional prototypical model parameter; ϵ_d and ϵ_κ are Gaussian-distributed zero-mean random variables with standard deviations $0.1/3$ and $0.05/3$, respectively. An example of two such classes of SDWs is defined by $(\bar{d}_1, \bar{\kappa}_1, \bar{a}_1) = (1.3,1.5,-0.1)$ and $(\bar{d}_0, \bar{\kappa}_0, \bar{a}_0) = (1.0,1.0,0.1)$ corresponding to the blue dash-dot and red solid curves in Fig. 2(a), respectively. It is more likely for the trajectories from class 0 to stay above, rather than below, the horizontal line with $\mathbf{x}_t = 0$. The opposite holds for class 1. This is because the asymmetry parameters of these two classes take their values with opposite signs. As a result, the classification task can be well accomplished by a classifier based on simple features directly extracted from the signal—in this case the overall trajectory mean. Figure 3 illustrates a contrasting task in which two classes of SDWs are defined by $(\bar{d}_1, \bar{\kappa}_1, \bar{a}_1) = (1.2,1.5,0)$ and $(\bar{d}_0, \bar{\kappa}_0, \bar{a}_0) = (1.0,1.5,0)$ [see the blue dotted and red dashed curves, respectively, in Fig. 2(a)]. As the mean asymmetry parameter is set to zero for both classes, the overall trajectory mean fluctuates around zero across the trajectories

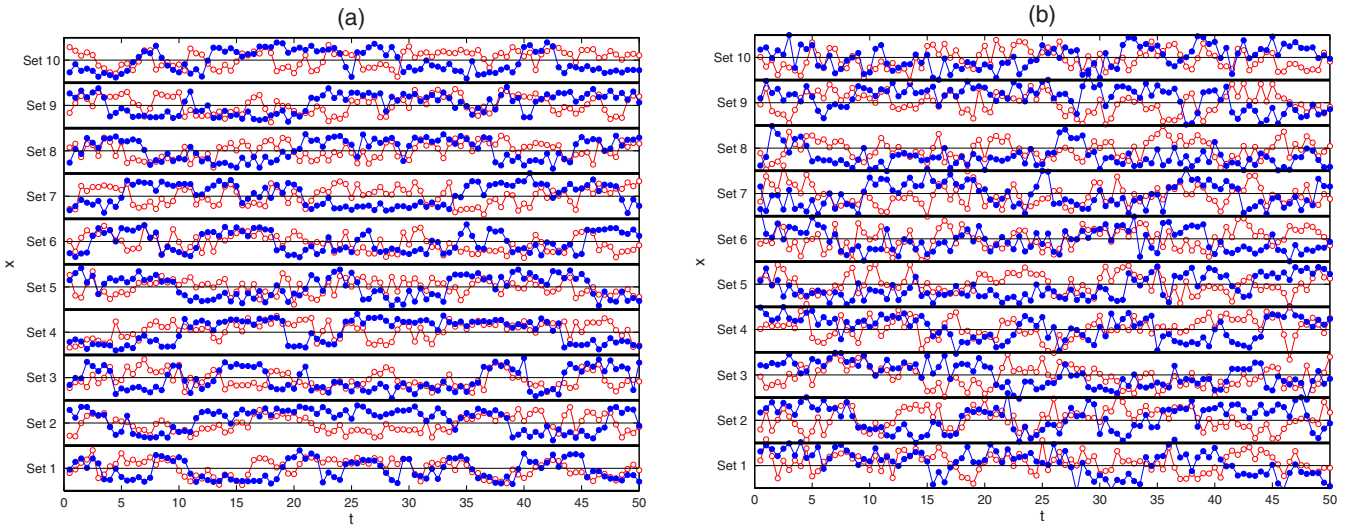

FIG. 3. (a) The observed trajectories of ten example stochastic double-well systems from each of two clusters in Fig. 2 (red curves with empty circles vs blue curves with filled circles). The range of each subpanel's vertical axis is scaled to $[-2.5, +2.5]$. The intersample interval (ISI) is 0.5, and the variance σ^2 of Gaussian distributed observation noise is 0.04. (b) The same as in (a) but for $\sigma^2 = 0.36$.

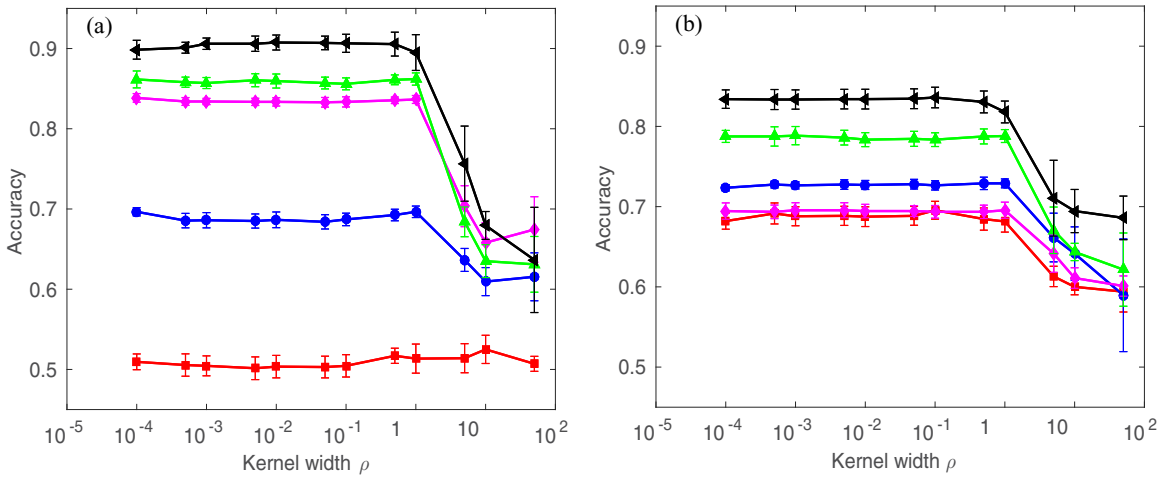

FIG. 4. Classification performance as function of log kernel width (i.e. $\log_{10}\rho$) using Learning in Model Space (LiMS) method to classify Partially Observed GnRH Models. Panel (a): The inter-sample intervals (ISI) is fixed 75 while the noise standard deviation varies across the set {0.1,0.03,0.01,0.005,0.001} (red squares, blue circles, magenta diamonds, green triangles, and black left triangles, respectively). All observations were sampled on a fixed regular grid over a 7.5h time window. Panel (b): The standard deviation of observation noises is fixed to 0.3 while the ISI value varies across the set {15,30,45,75,90} (red squares, blue circles, magenta diamonds, green triangles, and black left triangles, respectively). The observation times are random and the ISI-values given are the expected value. Values of the kernel width hyper-parameter for the LiMS classifier were taken from {0.0001,0.0005,0.001,0.005,0.01,0.05,0.1,0.5,1,5,10,50}. Note that the kernel width hyper-parameter ρ of LiMS and KME classifier is introduced in Eq. (19) as the scale parameter of Gaussian kernels.

in each of these two classes. We thus hypothesize that in such cases, the proposed classification LiMS framework will be superior to classification based on direct signal based features.

VI. EXPERIMENTS

A. Practical issues

In this section we discuss a number of practical issues related to testing the LiMS, KME, and PPK classifiers:

(i) *Does the input of a distributional classifier need to be normalized?* For the task of classifying PODS, the actual input is the posterior distribution over parameter vectors. In our setting, it includes a set of posterior probabilities defined on a grid of parameter vectors. For PPK classifiers, only those probabilities are used and thus there is no need for normalization. For the other two classifiers, however, we use parameter vectors (on the grid) together with the corresponding posterior probabilities. Moreover, the parameter vectors are involved in the classification via a spherical kernel function that is defined on the product of two parameter grids. Therefore, we normalize the parameter grid to vary in each dimension from 0 to 1. Of course, the original parameter values associated with grid points will be preserved.

(ii) *How to initialize the classifier's parameters for gradient-based training?*

We implement all three classifiers in the KLR framework. Hence, the PPK-based classifier parameter effectively weights the training examples, whereas in the case of LiMS and KME,

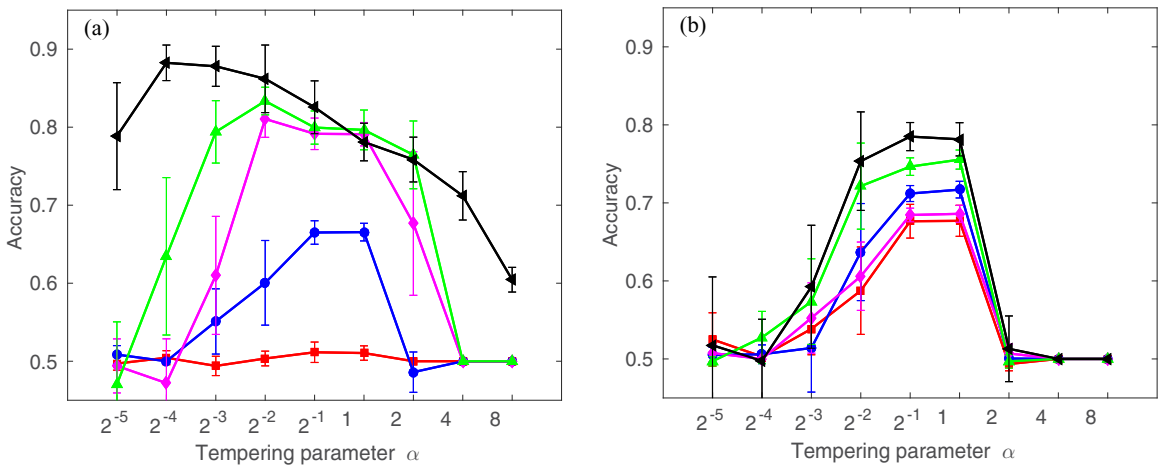

FIG. 5. As same as in Fig. 4 but for classification performance as function of log tempering parameter (i.e., $\log_2\alpha$) using probability product kernel (PPK) method. The tempering PPK hyperparameter took values from {1/32,1/16,1/8,1/4,1/2,1,2,4,8}. Note that the tempering hyperparameter α of PPK classifier is introduced in Eq. (32).

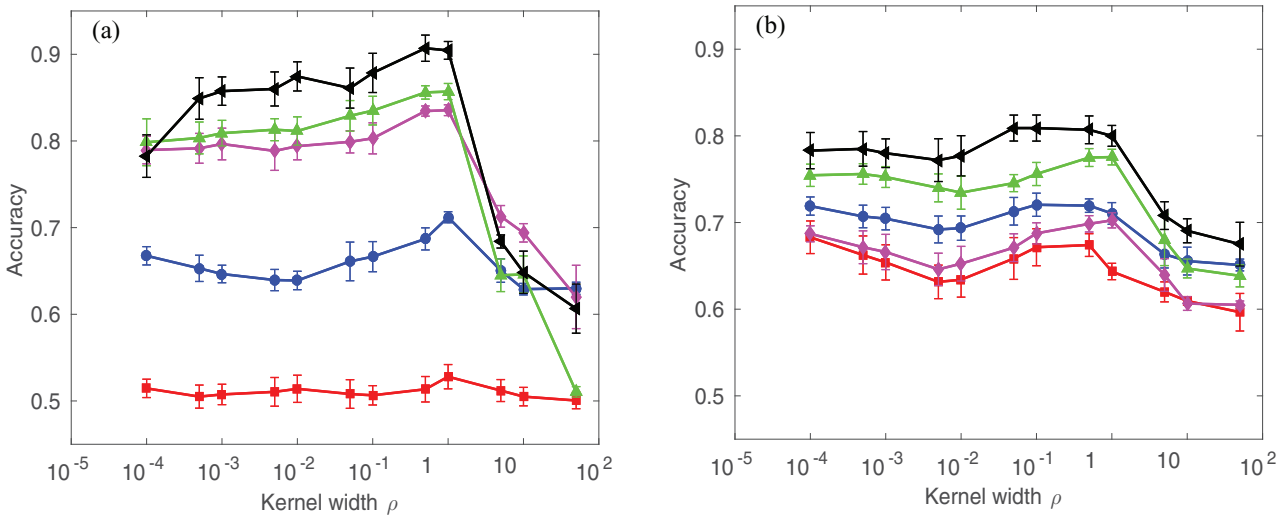

FIG. 6. Same as in Fig. 4 but for kernel mean embedding (KME) classifier.

the parameter puts weights on the model grid. In this work, all elements of the parameter vectors are initialized by drawing from Gaussian distribution with zero mean and unit variance. The parameters are then optimized through gradient descent as explained in Sec. III B. This procedure is repeated N^{init} times, resulting in N^{init} classifiers combined in flat ensemble outputting the average of the N^{init} predictive class probabilities (given a test input). We set $N^{\text{init}} = 15$.

(iii) For the binary classification tasks in this work, we first generate the training and hold-out test sets with balanced class distribution, each containing 200 observation time series. Both classes from the training set are randomly subsampled (without replacement) to 45 time series (out of 100), yielding a training batch of 90 time series. This is repeated $N^{\text{rand}} = 10$ times. We then report the mean ($\pm$ standard deviation) classification performance on the test set across the N^{rand} runs.

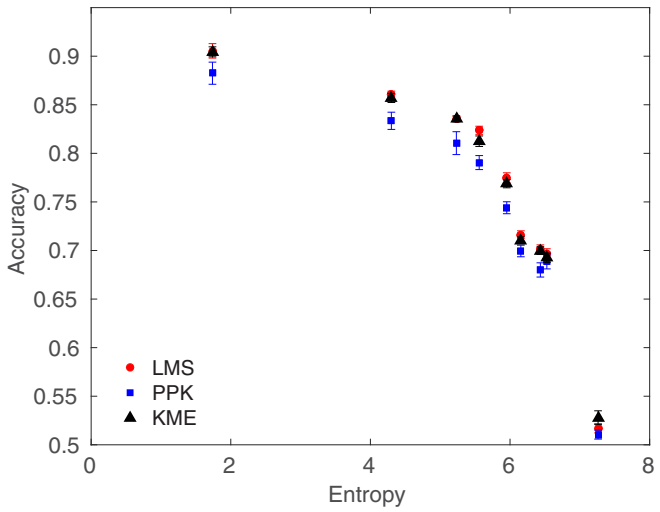

FIG. 7. Relationship between classification performance and model uncertainty measured by average posterior entropy. Each data point corresponds to one of 15 data sets in group 1, group 2, and group 3. The x- and y coordinate of each $\diamond$, $\square$, and $\circ$ point display the average posterior entropy and the accuracy for LiMS, PPK, KME classifiers, respectively.

B. GnRH signalling model

To conduct experiments with the classification task defined in Sec. V A, we generate two independent sets of GnRH models for training and testing (200 labeled models each). To that end we randomly sample 400 parameter vectors $\boldsymbol{\theta}_{\text{GnRH}} = (\log K_{d_{\text{TF}_1}}, \log K_{d_{\text{TF}_2}}, t_p)$ of the GnRH model. We consider log values of $K_{d_{\text{TF}_1}}$ and $K_{d_{\text{TF}_2}}$ since their permissible range extends over several magnitudes. Each of the three model parameters are sampled from the corresponding Gaussian distribution truncated to the permissible range. For each parameter, the mean and standard deviation of the untruncated Gaussian are set to the midpoint and radius, respectively of the permissible range (see Table I). The parameter vectors are then labeled as class 0 (normal conditions) or class 1 (abnormal conditions) as described in Sec. V A (see the right panel of Fig. 1).

As the task is to classify PODS, we generate a variety of observation time series with different observation settings (number of observations, observation times, and observational noise level). To sample observations from the GnRH model we first simulate GnRH (8-h window) and record the observable trajectory [GSU] at six different pulse frequencies. This results in a six-dimensional [GSU] trajectory with a time resolution of 1 min. Throughout the experiments, the initial values of state variables in GnRH model are fixed but the trajectory over the first half hour is discarded. This ensures that the transient behavior has been ignored and only the attractor part of individual trajectories is used for sampling observations and thus the initialization of the GnRH model has little influence on inferring the underlying model from observations [29]. Given a simulated [GSU] trajectory, we generate 15 observation sets using different pairs of observation noise level σ and the intersample interval (ISI). The observation sets are organized in three groups (five sets in each group):

Group 1. In each of the five observation sets, observations were sampled regularly every ISI = 75 min over 7.5 h, yielding six observation times. The level σ of observation noise in the five observation sets was set to 0.1, 0.03, 0.01, 0.005, and 0.001. Hence, the observation sets in this group correspond to the partially observed GnRH model with five different levels of model uncertainty controlled by σ.

TABLE II. Sign-rank tests for comparing the performance of LiMS, KME, and PPK classifiers at different levels of model uncertainty with the following one-sided hypotheses: (H1) LiMS outperforms KME; (H2) LiMS outperforms PPK; and (H3) KME outperforms PPK. The p values from these tests are given in columns 4–6 and all p values smaller than 0.05 are highlighted in bold font. The level of model uncertainty is measured by (average) posterior entropy (column 1). The corresponding observation noise level σ and the intersample interval ISI values are given in columns 2 and 3, respectively.

Entropy	(σ, ISI)	H1	H2	H3
1.7	(0.001, 75)	0.39	**0.01**	**0.01**
4.3	(0.005, 75)	0.22	**0.00**	**0.02**
5.2	(0.01, 75)	0.49	**0.02**	**0.01**
5.6	(0.03, 15)	**0.01**	**0.00**	**0.00**
6.0	(0.03, 30)	0.17	**0.00**	**0.00**
6.2	(0.03, 45)	0.09	**0.01**	**0.02**
6.4	(0.03, 75)	0.36	**0.00**	**0.00**
6.5	(0.03, 90)	0.12	0.07	0.29
7.3	(0.1, 75)	0.95	0.15	**0.01**

Group 2. Unlike in group 1, the five observation sets in this group are generated by fixing the observation noise to $\sigma = 0.03$ and varying the number of regularly spaced observation times within the 7.5-h window. In particular, the five observation sets contained 5, 6, 10, 15, and 30 observation times with ISI = 90, 75, 45, 30, and 15, respectively. In this case, the model uncertainty is controlled by the sparsity of observations.

Group 3. The σ and ISI values are as same as in group 2, but the observation times are placed randomly with uniform distribution over the 7.5-h window.

In order to apply a distributional classifier for the classification task, each observation set is represented by the corresponding posterior distribution over the GnRH models. Recall that in this work we approximate posteriors on a finite grid: $\log_{10}(K_{d_{\mathrm{TF}_1}}) = -2.3 + 1.4\frac{i}{41}$, $\log_{10}(K_{d_{\mathrm{TF}_2}}) = -2.1 + 1.4\frac{i}{41}$, $i = 0,1,2,\ldots,40,41$, and $t_p \in \{5,6,7,8,9,10\}$. The finite-grid encodes our prior knowledge about the biologically permissible parameter ranges. As the classes are discriminated by $K_{d_{\mathrm{TF}_1}}$ and $K_{d_{\mathrm{TF}_2}}$, the inferred posteriors are marginalized over t_p.

Experiment 1

In this experiment, we investigated the interplay between the classification performance of the three classifiers (LiMS, PPK, KME) and the level of model uncertainty. In particular, we first used group 1 data to study the relation between the accuracy and the level of observation noise [panel (a) in Figs. 4–6]. We then used group 2 and group 3 data to evaluate the relation between the accuracy and frequency of observations. The results are presented in panel (b) of Figs. 4–6. Only performance curves for group 3 are shown as the results for group 2 are very similar to those for group 3. Finally, we used all groups to assess the interplay between the accuracy and model uncertainty quantified by the average posterior entropy (Fig. 7).

Figure 4 shows results for the LiMS classifier. In each panel, we plot the testing accuracy against the log kernel width (i.e., $\log_{10}\rho$) for different σ values [panel (a)], or different ISI values [panel (b)]. Figure 4 shows in general that the performance increases with decreasing kernel width until a saturation level is reached (approximately) at $\rho = 1$. Recall that parameter vectors were normalized to lie within the unit cube. Kernel widths substantially larger than 1 introduce a strong model bias that leads to performance degradation. The only exception is the case where the model uncertainty is so large that the classifier performs as bad as random guess [see the curve corresponding to ISI = 75 and $\sigma = 0.1$ in panel (a)]. On the other hand, it is interesting to observe that the performance is quite robust with respect to kernel width variations below the critical scale of 1. Figure 4 also shows that the performance increases (almost) monotonically with decreasing ISI or σ, which confirms the hypothesized relationship between classification performance and model uncertainty.

The results for PPK and KME classifier are displayed in Figs. 5 and 6, respectively. Together with Fig. 4, they show that the character of the interplay between classification performance and model uncertainty is very similar for all three classifiers. Kernel parameters of LiMS and KME classifiers can be related to each other and from this point of view, the LiMS classifier appears to be more robust to variations in the kernel parameter, which is a desirable property. However,

TABLE III. The LiMS's task performance for classifying partially observed GnRH model when using three different inferential GnRH model structures (i.e., $M1$, $M2$, and $M3$ described in Sec. V A) to infer the input posterior distributions from the [GSU] time series. For these inferential models, their corresponding mean performance ($\pm$ standard deviation) obtained from nine different time series data sets are summarized in columns 3–5, respectively. The observation settings of these data sets are given in columns 1 and 2, where σ denotes the observation noise level and ISI the sampling frequency.

Data sets	(σ, ISI)	$M1$	$M2$	$M3$
Group 1	(0.001, 75)	0.91 ± 0.02	0.88 ± 0.03	0.90 ± 0.01
	(0.01, 75)	0.84 ± 0.01	0.84 ± 0.01	0.83 ± 0.01
	(0.1, 75)	0.52 ± 0.01	0.54 ± 0.01	0.54 ± 0.01
Group 2	(0.03, 90)	0.82 ± 0.00	0.81 ± 0.01	0.81 ± 0.01
	(0.03, 30)	0.74 ± 0.01	0.73 ± 0.01	0.72 ± 0.01
	(0.03, 45)	0.71 ± 0.01	0.70 ± 0.02	0.69 ± 0.01
Group 3	(0.03, 90)	0.83 ± 0.02	0.82 ± 0.02	0.82 ± 0.01
	(0.03, 30)	0.69 ± 0.01	0.71 ± 0.01	0.71 ± 0.01
	(0.03, 30)	0.68 ± 0.02	0.69 ± 0.01	0.68 ± 0.02

TABLE IV. Left: The specification of two classes of partially observed stochastical double-well systems in three classification tasks by their respective prototypical model parameters. Right: The specification of two groups of observation sets generated for each of three tasks by their respective observation noise level σ and intersample interval ISI.

($\bar{d}$,$\bar{\kappa}$,$\bar{a}$)	Class 1	Class 0	(σ, ISI)	Group 1	Group 2
Task 1	(1.0, 1.0, −0.1)	(1.3, 1.5, 0.1)	Set 1	(0.3, 0.5)	(0.3, 0.5)
Task 2	(1.0, 1.5, 0)	(1.3, 1.5, 0)	Set 2	(0.4, 0.5)	(0.3, 1.0)
Task 3	(1.0, 1.5, 0)	(1.2, 1.5, 0)	Set 3	(0.6, 0.5)	(0.3, 1.25)

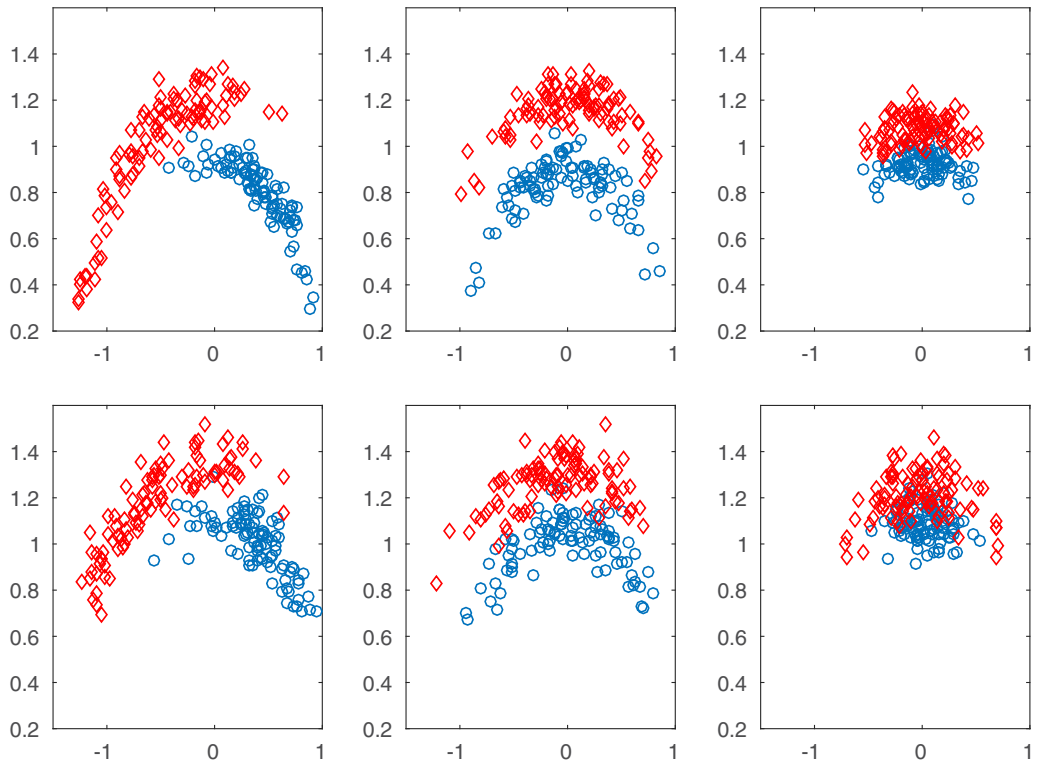

FIG. 8. Scatter plot of the mean and standard deviation pairs (μ_y, γ_y) computed for the time series $\{y_t\}$ observed (1) from different tasks [from left to right: task 1, task 2, and task 3] and (2) with different (σ, ISI) settings [from top to bottom: (0.2,0.5), and (0.6,0.5)]. The data points in the scatter plots from class 1 and class 0 are displayed as red diamonds and blue circles, respectively.

the role of the tempering kernel parameter in a PPK classfier is very different and hence no direct comparison of performance stability with varying kernel parameter can be made with LiMS and KME classifiers.

Kernels in KME and PPK classifiers effectively smooth and temper, respectively, the input posterior distributions. As in the case of LiMS classifier, for KME the optimal kernel width is around 1 (parameter vectors are normalized to lie within a unit cube). For PPK, it seems that in most cases, high classification performance is obtained when the posterior distributions are not (or just slightly) tempered. The only exception is group 1 PPK curve corresponding to $\sigma = 0.0001$ and ISI $= 150$, where the tempering flattens the model posteriors.

Figure 7 shows the classification performance as a function of model uncertainty for the three PODS classifiers (LiMS, KME, and PPK). For each of 15 data sets, the level of model uncertainty is computed by averaging entropies of model posterior distributions inferred from the individual observed time series. The performance is quantified through the accuracy at the kernel parameter determined individually for each classifier and each data set on the validation data. For the LiMS classifier, we chose $\rho = 0.5$ as the overall "optimal" kernel width. For the KME classifier, its optimal kernel width is chosen as $\rho = 1.0$ for group 1 and 2 data and as $\rho = 0.5$ for group 3 data. In the case of the PPK classifier, we chose $\alpha = 0.5$ for group 2 data and $\alpha = 1.0$ for group 3 data. For group 1 data, however, the PPK's optimal tempering parameter decreases with σ, that is, $\alpha = 0.5$ for $\sigma = 0.1$ and $\sigma = 0.03$, $\alpha = 0.25$ for $\sigma = 0.01$ and $\sigma = 0.005$, and $\alpha = 0.0625$ for $\sigma = 0.001$. Recall that for each of the 15 observation sets, we have ten performance measures obtained on ten resampled training or hold-out sets. We combine all performance and uncertainty measures corresponding to the same ISI and σ (regardless of whether or not the observation times are random) into a single set. This results in nine sets of (uncertainty, performance) values. For each set, the corresponding average posterior entropy, observation noise level σ, and intersample interval ISI are given in the first two columns of Table II. In Fig. 7 the means and standard deviations of the performance measures are plotted against the corresponding average posterior entropy. The plot shows a clear dropoff in classification performance at high model uncertainty (of about 5 nats, where nat represents a unit of entropy, based on natural logarithms). In this respect, there is no significant difference among the three classifiers. However, for low and moderate model uncertainty levels both LiMS and KME outperform PPK. LiMS and KME also have comparable classification performance. In Table II, the p values from sign-rank statistical tests are given for the following one-sided hypotheses: (H1) LiMS outperforms KME; (H2) LiMS outperforms PPK; and (H3) KME outperforms PPK. The p values here mean the probability for the corresponding (one-sided) hypothesis being true just by chance.

Experiment 2

In this experiment we investigate whether the performance of classifying partially observed GnRH models would be impaired if simpler reduced complexity GnRH model structures $M2$ and $M3$ of Sec. V A were used to infer the input posterior

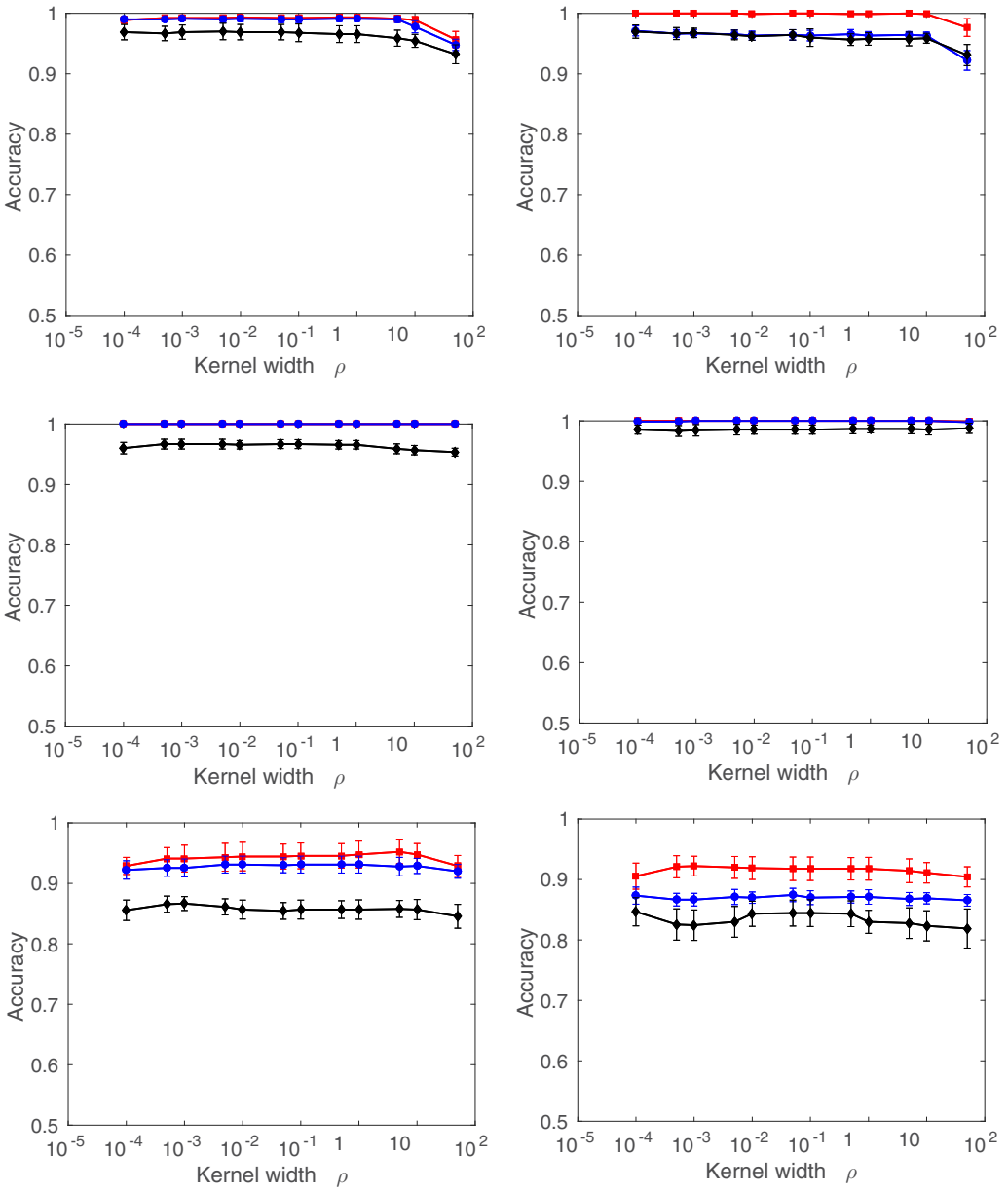

FIG. 9. Classification performance as function of log kernel width (i.e., $\log_{10}\rho$) using LiMS classifier to classify partially stochastic double-well systems for different tasks (from top to bottom: task 1–task 3) and for different observation settings [left: (σ,ISI) = (0.3,0.5), (0.4,0.5), and (0.6,0.5) with red squares, blue circles, and black diamonds, respectively; right: (σ,ISI) = (0.3,0.5), (0.3,1.0), and (0.3,1.25) with red squares, blue circles, and black diamonds, respectively)].

distributions representing observation sets generated from the full model $M1$.

For each time series data set, we evaluated the LiMS performance when using the $M1$- (as a reference), $M2$- and $M3$-generated posterior distributions representing the observation sequences. The results, summarized in Table III, show that the performance is closely comparable for all inferential model structures $M1$–$M3$, for all observation sets. Recall that two classification GnRH classes differ only in their frequency-response characteristics that are completely determined by the $K_{d_{\mathrm{TF}_1}}$ and $K_{d_{\mathrm{TF}_2}}$ values. All three model structures $M1$–$M3$ include compartment C3 which modulates the observable model output [GSU] via Eq. (45). Moreover, the dynamics of [GSU] is controlled by $K_{d_{\mathrm{TF}_1}}$ and $K_{d_{\mathrm{TF}_2}}$. Our results confirm one of the key points of this study: For classification of PODS via the learning in the model space framework, it is not necessary for the inferential model structure to be a perfect model of the underlying dynamical system generating the data, as long as the reduced complexity inferential model structure captures the essential characteristics needed for the given classification task.

C. Double-well model

For partially observed stochastical double-well systems (SDWs), the task is to classify posterior distributions over the models accessible through parameter vectors (a,d,κ). Recall that a is the asymmetry parameter, d is the well location parameter, and κ represents the dynamical noise level. Also recall that the two classes of SDWs involved in our

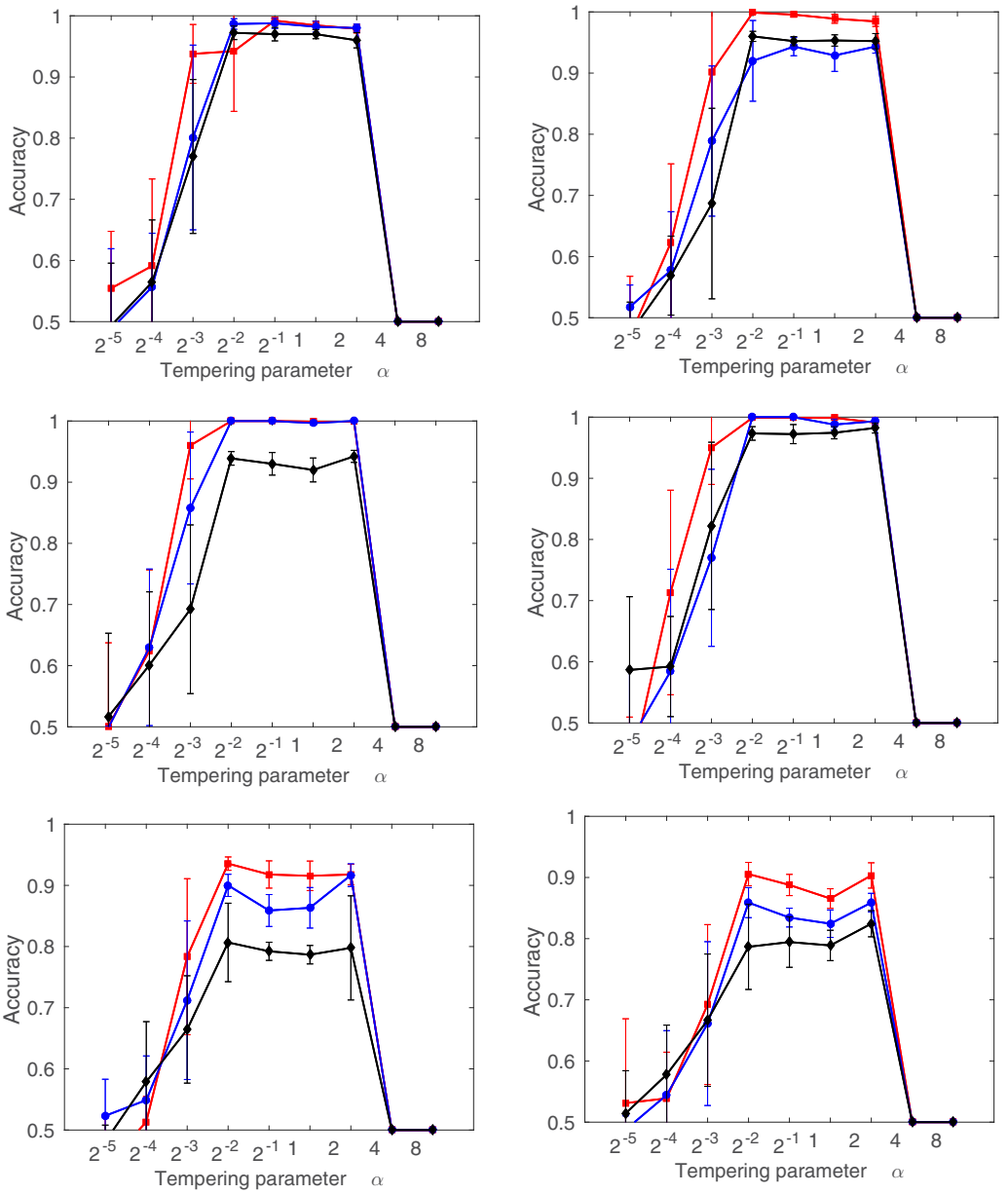

FIG. 10. The same as in Fig. 9 but for classification performance as function of log tempering parameter (i.e., $\log_2 \alpha$) using probability product kernel (PPK) method.

experiments are defined through two class-conditional Gaussian distributions in the parameter space: $(\bar{d}_1 + \epsilon_d, \bar{\kappa}_1 + \epsilon_\kappa, \bar{a}_1)$ for class 1 and $(\bar{d}_0 + \epsilon_d, \bar{\kappa}_0 + \epsilon_\kappa, \bar{a}_0)$ for class 0, where $(\bar{d}_1, \bar{\kappa}_1, \bar{a}_1)$ and $(\bar{d}_0, \bar{\kappa}_0, \bar{a}_0)$ denote the class-conditional prototypical model parameter; ϵ_d and ϵ_κ are Gaussian-distributed zero-mean random variables with standard deviations 0.1/3 and 0.05/3, respectively.

To compare our LiMS classifier with KME and PPK classifiers, we define a hierarchy of three tasks of increasing complexity, denoted by task 1–task 3 (see Table IV). Furthermore, to investigate the relation between the level of model uncertainty and classification performance, for each of the three tasks, we generate two groups of observation sets (denoted by group 1 and group 2). Each group consists of three observation sets with varying degrees of model uncertainty. As in the GnRH experiment, the model uncertainty level induced by each observation set is determined by the corresponding observation noise level σ and the intersample interval ISI. The increase of uncertainty level in group 1 and group 2 is modulated by increasing σ and ISI, respectively (see Table IV). For both groups, the time series in each observation set were sampled at regularly spaced observation times (with intersample interval ISI) within the time interval [0, 50].

As we adopt a finite-grid approximation approach to compute the model posteriors, the parameter space Θ is discretized as follows: $d \in \{0.1, 0.2, \ldots, 1.9, 2.0\}$, $\kappa \in \{0.1, 0.2, \ldots, 1.9, 2.0\}$, and $a \in \{-0.2, -0.1, 0, 0.1, 0.2\}$.

One may argue that, given the nature of the classification tasks outlined above, the mean μ_y and standard deviation γ_y of the observed time series $\{y_t\}$ can provide useful features for building a classifier solely operating in the signal space.

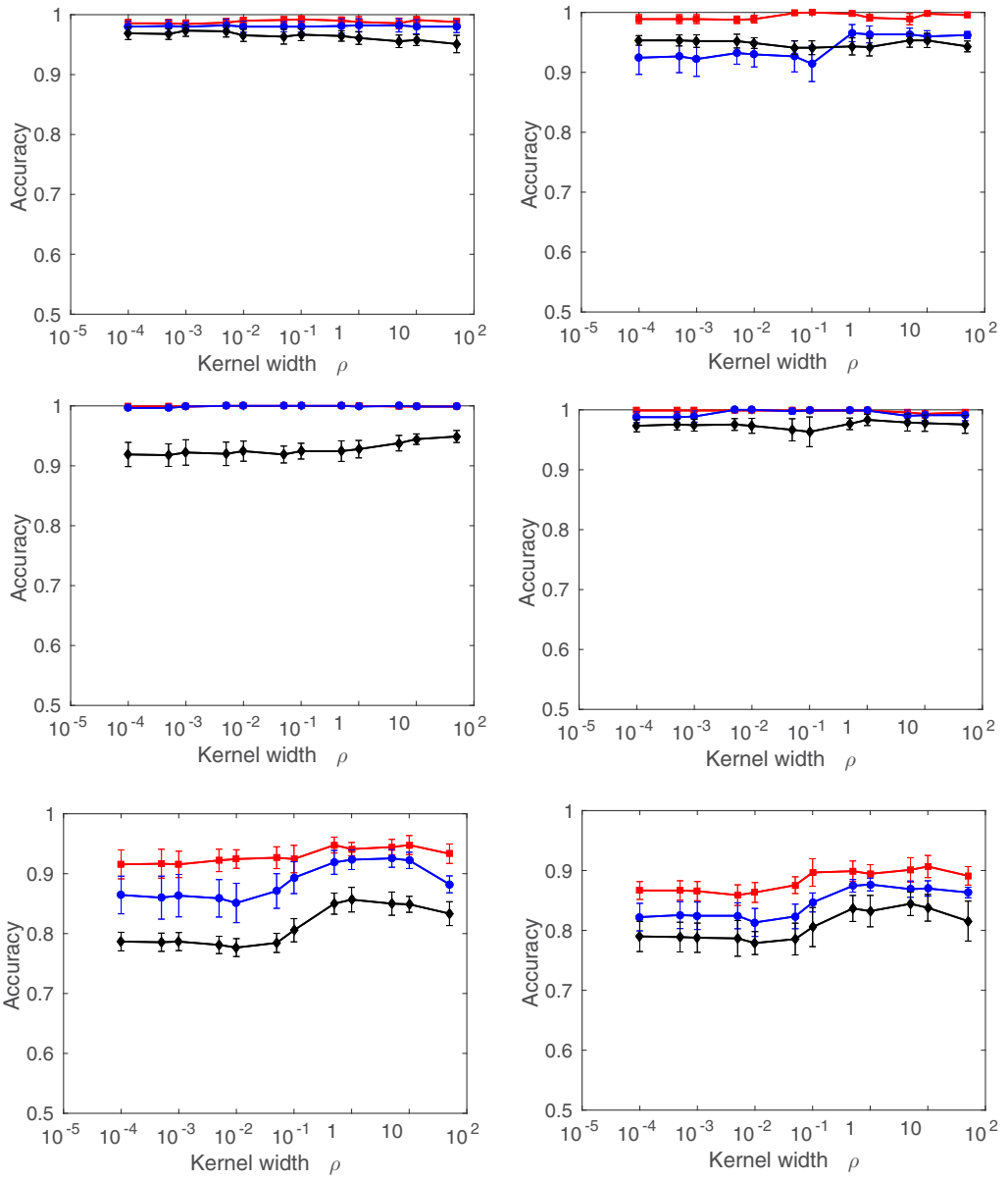

FIG. 11. The same as in Fig. 9 but for KME classifiers.

Such feature vectors (μ_y,γ_y) can also provide an insight regarding the task complexity. Figure 8 shows six scatter plots of (μ_y,γ_y) for tasks 1, 2, and 3 (left, middle, and right column, respectively) and for $(\sigma,\mathrm{ISI}) = (0.3,0.5)$ and $(\sigma,\mathrm{ISI}) = (0.6,0.5)$ (upper and lower row, respectively). The class labels are indicated by coloured symbols (red diamonds for class 1 and blue circles for class 0). For task 1, the asymmetry parameter a is class dependent and Fig. 8 shows that in this case, the two classes can be separated simply by using the time series' means μ_y. For example, a positive value of a would cause the means μ_y of time series from the corresponding class to be biased towards a positive value and vice versa. In task 2, $a = 0$ for both classes and the means μ_y can no longer separate the two classes. However, classification is still possible in the joint space (μ_y,γ_y). By gradually reducing the difference between the two classes in terms of the dynamical noise level κ, the classes can be brought closer together in the (μ_y,γ_y) space in a controlled manner. To tease out possible advantages of the learning in the model space framework, in all SDW experiments we also employ a signal-space baseline KLR classifier (bKLR) solely operating on (μ_y,γ_y).

Experiment 1

Figure 9 shows the LiMS performance as a function of kernel width ρ for tasks 1–3 and for different combinations of σ and ISI values. In particular, in plots on the left the ISI is fixed to 0.5 and $\sigma = 0.3,0.4,0.6$; in plots on the right the σ is fixed to 0.3 and $\mathrm{ISI} = 0.5,1,1.25$. Overall, the classification performance remains robust over a fairly large interval of intermediate ρ values ranging from 0.01 to 1.0. Naturally, there is a drop in classification performance at very large kernel width $\rho = 10$. Further, as expected, the performance

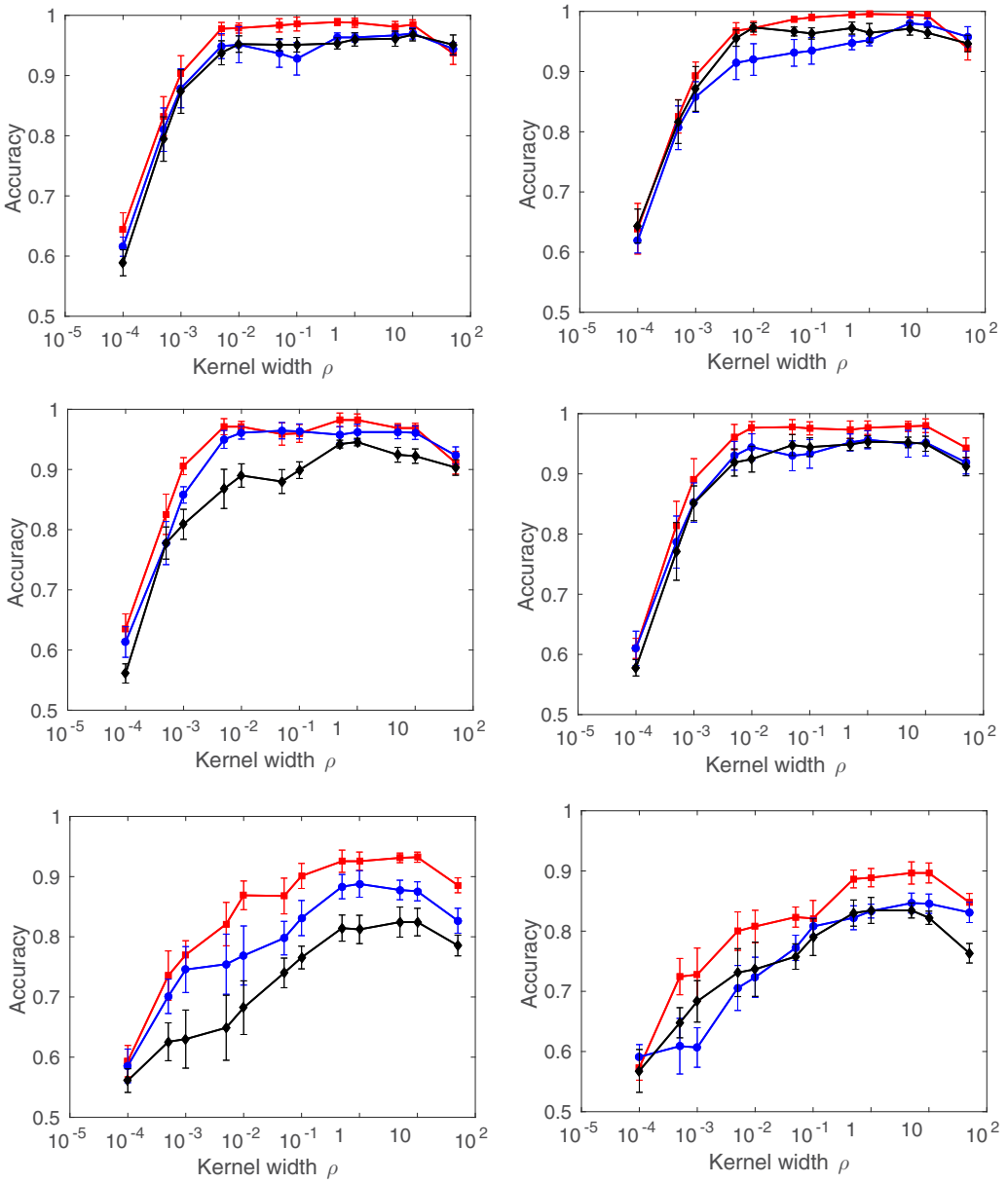

FIG. 12. The same as in Fig. 9 but for bKLR classifiers.

decreased monotonically with increasing σ or ISI for all intermediate kernel widths. These findings match observations made in the GnRH experiments. Figure 10 shows that the PPK classifier maintains its maximum performance over an interval of intermediate tempering parameter values ranging from $\alpha = 2^{-3}$ to $\alpha = 2$. Recall that for GnRH models, the PPK classifier attained the best performance for $\alpha \leqslant 1.0$. Values of $\alpha > 1$ effectively make the input posterior distributions over the models more peaked prior to classification. Unlike in the GnRH experiments, in general the KME classifier retains its best performance for larger kernel widths $\rho \geqslant 0.5$ (see Fig. 11). Figure 12 shows that the performance of the baseline bKLR classifier increased steadily with the kernel width, achieving its best performance over a range of large kernel widths. For subsequent analysis, we chose (using the validation data) $\rho = 0.05$, $\alpha = 2$, $\rho = 1$, and $\rho = 1$ as the overall kernel parameters for the LiMS, PPK, KME, and bKLR classifiers, respectively.

To compare the four classifiers in a statistical manner, we tested six different one-sided hypotheses: (H1) LiMS outperforms KME; (H2) LiMS outperforms PPK; (H3) KME outperforms PPK; (H4) LiMS outperforms bKLR; (H5) KME outperforms bKLR; and (H6) PPK outperforms bKLR. In particular, hypotheses H4–H6 address the question what kind of learning in the model space classifiers can outperform the baseline classifier operating in the signal space. The results are summarized in Table V. All p values smaller than 0.15 are highlighted in bold font. Table V shows that for task 2, all three posterior-based classifiers clearly outperform the bKLR classifier. For task 3, LiMS and KME still outperform bKLR. The results for the simplest task 1 indicate that LiMS would have the upper hand against bKLR but the overall trend is not

TABLE V. Sign-rank tests for comparing the classification performance between LiMS, KME, PPK, and bKLR classifiers in the three tasks of classifying partially observed double-well systems, using the following one-sided hypotheses: (H1) LiMS outperforms KME; (H2) LiMS outperforms PPK; (H3) KME outperforms PPK; (H4) LiMS outperforms bKLR; (H5) KME outperforms bKLR; and (H6) PPK outperforms bKLR. The p values from these tests are given in columns 3–8 and all p values smaller than 0.15 are highlighted in bold font. The level of model uncertainty is measured by (average) posterior entropy (column 2).

Task	Entropy	H1	H2	H3	H4	H5	H6
Task 1	4.564	**0.00**	**0.00**	**0.02**	**0.07**	0.73	0.99
	4.634	**0.01**	**0.10**	0.89	0.48	1.00	0.93
	4.654	0.25	**0.04**	0.30	**0.00**	**0.01**	**0.01**
	4.656	0.67	**0.01**	**0.01**	**0.03**	**0.09**	0.89
	4.756	**0.13**	**0.06**	0.62	0.22	0.39	0.47
Task 2	4.561	0.50	**0.03**	**0.09**	**0.00**	**0.00**	**0.00**
	4.682	0.50	**0.06**	**0.13**	**0.00**	**0.00**	**0.00**
	4.693	0.50	**0.13**	0.50	**0.00**	**0.01**	**0.01**
	4.707	0.50	1.00	1.00	**0.00**	**0.00**	**0.00**
	4.835	**0.00**	**0.00**	0.94	**0.00**	1.00	0.73
Task 3	4.659	**0.01**	**0.00**	**0.14**	**0.01**	**0.07**	0.36
	4.762	0.82	**0.10**	**0.03**	**0.00**	**0.00**	**0.02**
	4.775	0.22	**0.05**	0.31	0.40	0.73	0.78
	4.837	**0.13**	**0.06**	0.16	**0.00**	**0.02**	**0.04**
	5.026	0.73	**0.13**	**0.08**	**0.00**	**0.01**	0.37

clear. Indeed, in task 1 the two classes can be conveniently separated in the signal space. This analysis shows the overall superiority of LiMS (but not KME) over PPK. Interestingly enough, we observed the same general trend of decreasing classifier performance with increasing model uncertainty in the input model posteriors.

Experiment 2

Finally, we study to what degree can the use of a simpler model to obtain representative model posteriors hamper the classifier performance, provided the observations are generated by a much more complex model, yet the simpler model already embodies characteristics needed to perform the given classification task (see Sec. I). In particular, we form an extended task 1, task 1e, in which time series in the observation sets were generated by complex stochastic multiwell systems with multimodal structure of the equilibrium distribution that can approximated (for the purposes of classification) by SDW systems (see Fig. 2). The performance of LiMS classifier in task 1e, reported in column 2 in Table VI, was compared with task 1 (column 3 of the same table). The p values for the one-sided hypothesis stating that a better classification performance can be obtained in task 1e than in task 1 are given in column 4. Overall, the performance in task 1e is as good as in task 1. This confirms analogous findings in the GnRH experiment, where the use of simplified models, well aligned with the classification task, did not hamper the classification performance, even though the observation sequences were generated by much more complex models (see Sec. VI B, Experiment 2).

TABLE VI. Comparison of classification performance between two different classes of data-generating SDW systems: multiwell systems vs double-well systems. Note that double-well systems are the inferential model used in both cases.

(σ, ISI)	Task 1e	Task 1	p value
(0.3, 0.5)	0.992 ± 0.005	0.996 ± 0.004	1.00
(0.4, 0.5)	0.986 ± 0.006	0.987 ± 0.003	0.82
(0.6, 0.5)	0.978 ± 0.009	0.974 ± 0.008	0.09
(0.3, 1.0)	0.984 ± 0.004	0.996 ± 0.005	1.00
(0.3, 1.25)	0.970 ± 0.006	0.991 ± 0.005	1.00

VII. DISCUSSION AND CONCLUSION

In this paper, we have presented a general learning in the model space (LiMS) framework for classifying partially observed dynamical systems. The key ingredient of this framework is the use of posterior distributions over models to represent the individual observation sets, taking into account in a principled manner the uncertainty due to both the generative (observational and/or dynamic noise) and observation (sampling in time) processes. This is in contrast to the existing learning in the model space classification approaches that use model point estimates to represent data items. Another key ingredient of our approach is a distributional classifier for classifying posterior distributions over dynamical systems.

We evaluated this classifier on two test beds, namely a biological pathway model and a stochastic double-well system. Empirically the classifier clearly outperforms the classifier based on probability product kernel (PPK)—a state-of-the-art kernel method for classifying distributions. Moreover, its performance is comparable with a recent distributional classification method based on kernel mean embedding. We derived a deep connection linking those three seemingly diverse approaches to distributional classification and provided a plausible explanation concerning superiority of the proposed classifier over the PPK classifier.

The experiments show a clear relation between model uncertainty and classification performance. As expected, the performance drops with increasing model uncertainty. Principled treatment of model uncertainty in the learning in the model space approach is crucial in situations characterized by non-negligible observational noise and/or limited observation times. To illustrate this point further we also trained a baseline classifier that, given the observed time series, completely ignores the model uncertainty and instead of posterior distribution only employs the MAP point estimate of the model parameter. As all the other classifiers, the baseline classifier (referred to as MAP) is also implemented in the KLR framework.

We compared the three posterior based classifiers with the MAP classifier using both test beds. The comparison follows the philosophy of comparing baseline classifier (bKLR) with the distributional classifiers in the SDW experiment (columns 6–8 in Table V). In particular, in the GnRH experiment, we tested three hypotheses (distributional classifier outperforms MAP) at nine uncertainty levels (see column 1 in Table II). Both LiMS and KME classifiers outperform (in the mean) the MAP classifier in all, except for one, uncertainty levels.

For LiMS and KME, this superiority is statistically significant ($p < 0.05$) in all cases except for the lowest and the two lowest uncertainty levels, respectively. This is to be expected, as at low uncertainty levels the posterior over the models can be reasonably approximated by the MAP model estimate. In contrast, PPK classifier outperforms the MAP classifier only at four uncertainty levels, with statistical significance obtained only at the three highest uncertainty levels. In the SDW experiment, the tests were performed at 15 uncertainty levels (see columns 1 and 2 in Table V). The LiMS, KME, and PPK classifiers outperform the MAP classifier at all 15, 11, and 7 uncertainty levels, with statistical significance obtained at 9, 4, and 4 uncertainty levels, respectively.

Crucially, we showed that the classifier performance would not be impaired when the model class used for inferring posterior distributions is much more simple than the observation-generating model class, provided the reduced complexity inferential model class captures the essential characteristics needed for the given classification task. This finding is potentially very significant for real-world applications. Although mechanistic models encode expert domain knowledge and are of huge importance in forward modeling (e.g., assessing response to drug at certain dosage), such models may be too complex for the inferential (inverse-task) purposes. Fortunately, much reduced model alternatives can be used in the learning in the model space framework if, as explained above, they already encode features important for the classification task. A semiautomated task-driven model simplification for learning in the model space framework is a matter for our future research.

The classification framework presented in this paper is a discriminative approach to model-based classification with a principled treatment of model uncertainty. In the literature, the "Bayesian model reduction" framework developed by Friston *et al.* [32] can be used to effectively generate class-conditional empirical priors over model parameters of dynamical causal models, given a training set of the labeled posterior distributions. To classify a new data set, the empirical priors of all classes can be used to compute their respective Bayesian model evidence which are subsequently used as scores to perform classification. This represents a generative approach to model-based classification in which model uncertainty is taken into account via Bayesian model reduction in the training phase and Bayesian model comparison in the testing phase. Further, Bayesian model reduction could also allow for simultaneously performing model-based classification with two or more structural models of different complexities, provided that all structural models have the same state space but with a different number of free parameters. In our work, the full and reduced models we have tested share the same free parameters but with different dimensionality of their state spaces.

ACKNOWLEDGMENTS

This work was supported by the EPSRC grant "Personalised Medicine Through Learning in the Model Space" (Grant No. EP/L000296/1). K.T.-A. gratefully acknowledges the financial support of the EPSRC via Grant No. EP/N014391/1.

[1] T. W. Liao, Pattern Recognit. **38**, 1857 (2005).
[2] Z. Xing, J. Pei, and E. Keogh, ACM SIGKDD Explor. Newsl. **12**, 40 (2010).
[3] N. Lesh, M. J. Zaki, and M. Ogihara, in *KDD'99: Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, edited by S. Chaudhuri and D. Madigan (ACM Press, San Diego, CA, 1999), p. 342.
[4] C. C. Aggarwal, in *KDD'02: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, edited by D. J. Hand, R. Keim, R. Ng, O. R. Zaiane, and R. Goebel (ACM Press, Edmonton, Alberta, Canada, 2002), p. 163.
[5] H. Sakoe and S. Chiba, IEEE Trans. Acoust., Speech, Signal Process. **26**, 43 (1978).
[6] E. Birney, IBM J. Res. Dev. **45**, 449 (2001).
[7] K. H. Brodersen, T. M. Schofield, A. Leff, C. S. Ong, E. I. Lomakina, J. M. Buhmann, and K. E. Stephan, PLoS Comput. Biol. **7**, e1002079 (2011).
[8] K. J. Friston, L. Harrison, and W. Penny, NeuroImage **19**, 1273 (2003).
[9] H. Chen, F. Tang, P. Tino, and X. Yao, in *KDD'13: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, edited by I. S. Dhillon, Y. Koren, T. E. Senator, P. Bradley, R. Parekh, J. He, R. L. Grossman, and R. Uthurusamy (ACM Press, Chicago, IL, 2013), p. 392.
[10] H. Chen, F. Tang, P. Tino, and X. Yao, in *IJCAI'2015: Proceedings of the 24th International Joint Conference on Artificial Intelligence*, edited by Q. Yang and M. Wooldridge (International Joint Conference on Artificial Intelligence, Buenos Aires, Argentina, 2015), p. 3387.
[11] F. Cuzzolin, in *Machine Learning for Vision-Based Motion Analysis*, edited by L. Wang, G. Zhao, L. Cheng, and M. Pietikainen (Springer, New York, 2011), p. 55.
[12] F. Cuzzolin and M. Sapienza, IEEE Trans. Pattern Anal. Mach. Intell. **36**, 1483 (2014).
[13] P. J. Moreno, P. Ho, and N. Vasconcelos, in *Advances in Neural Information Processing Systems 16*, edited by S. Thrun, L. K. Saul, and B. Schölkopf (MIT, Cambridge, MA, 2004), p. 1385.
[14] A. B. Chan and N. Vasconcelos, in *CVPR'05: Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2005*, edited by C. Schmid, S. Soatto, and C. Tomasi (IEEE Computer Society Press, Los Alamitos, CA, 2005), p. 846.
[15] A. B. Chan and N. Vasconcelos, in *CVPR'07: Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2007*, edited by B. Bhanu and N. Ratha (IEEE Computer Society Press, Minneapolis, MN, 2007), p. 1.
[16] T. Jebara, R. Kondor, and A. Howard, J. Mach. Learn. Res. **5**, 819 (2004).
[17] S. V. N. Vishwanathan and A. J. Smola, Int. J. Comput. Vision **73**, 95 (2006).

[18] A. Bissacco, A. Chiuso, and S. Soatto, IEEE Trans. Pattern Anal. Mach. Intell. **29**, 1958 (2007).
[19] K. Muandet, K. Fukumizu, F. Dinuzzo, and B. Schölkopf, in *Advances in Neural Information Processing Systems 25* (MIT, Cambridge, MA, 2012), p. 10.
[20] A. Smola, A. Gretton, L. Song, and B. Schölkopf, in *Algorithmic Learning Theory: 18th International Conference* (Springer, New York, 2007), p. 13.
[21] T. Jaakkola and D. Haussler, in *Advances in Neural Information Processing Systems 11* (MIT, Cambridge, MA, 1998), p. 487.
[22] M. Cuturi and A. Doucet, arXiv:1101.0673.
[23] M. Seeger, in *Advances in Neural Information Processing Systems 14* (MIT, Cambridge, MA, 2002), pp. 905–912.
[24] M. Girolami, Theor. Comput. Sci. **408**, 4 (2008).
[25] F. Dondelinger, M. Filippone, S. Rogers, and D. Husmeier, J. Mach. Learn. Res., Workshop Conf. Proc. **31**, 216 (2013).
[26] C. Archambeau, M. Opper, Y. Shen, D. Cornford, and J. Shawe-Taylor, in *Advances in Neural Information Processing Systems 20* (MIT, Cambridge, MA, 2008), p. 17.
[27] A. Golightly and D. J. Wilkinson, Computat. Stat. Data Anal. **52**, 1674 (2008).
[28] K. J. Friston, J. Mattout, N. Trujillo-Barreto, J. Ashburner, and W. Penny, NeuroImage **34**, 220 (2007).
[29] K. Trapeva-Atanasova, P. Mina, C. J. Caunt, S. P. Armstrong, and C. A. McArdle, J. R. Soc., Interface **9**, 170 (2012).
[30] L. A. Peletier and W. C. Troy, in *Spatial Patterns* (Springer, New York, 2001), p. 239.
[31] J. Honerkamp, *Stochastic Dynamical Systems* (VCH, New York, 1993).
[32] K. J. Friston, V. Litvak, A. Oswal, A. Razi, K. E. Stephan, B. C. M. van Wilj, G. Ziegler, and P. Zeidman, NeuroImage **128**, 413 (2016).