GRU-ODE-Bayes: Continuous modeling of sporadically-observed time series

Edward De Brouwer[*†]
ESAT-STADIUS
KU LEUVEN
Leuven, 3001, Belgium
edward.debrouwer@esat.kuleuven.be

Jaak Simm[*]
ESAT-STADIUS
KU LEUVEN
Leuven, 3001, Belgium
jaak.simm@esat.kuleuven.be

Adam Arany
ESAT-STADIUS
KU LEUVEN
Leuven, 3001, Belgium
adam.arany@esat.kuleuven.be

Yves Moreau
ESAT-STADIUS
KU LEUVEN
Leuven, 3001, Belgium
moreau@esat.kuleuven.be

Abstract

Modeling real-world multidimensional time series can be particularly challenging when these are *sporadically* observed (*i.e.*, sampling is irregular both in time and across dimensions)—such as in the case of clinical patient data. To address these challenges, we propose (1) a continuous-time version of the Gated Recurrent Unit, building upon the recent Neural Ordinary Differential Equations (Chen et al., 2018), and (2) a Bayesian update network that processes the sporadic observations. We bring these two ideas together in our GRU-ODE-Bayes method. We then demonstrate that the proposed method encodes a continuity prior for the latent process and that it can exactly represent the Fokker-Planck dynamics of complex processes driven by a multidimensional stochastic differential equation. Additionally, empirical evaluation shows that our method outperforms the state of the art on both synthetic data and real-world data with applications in healthcare and climate forecast. What is more, the continuity prior is shown to be well suited for low number of samples settings.

1 Introduction

Multivariate time series are ubiquitous in various domains of science, such as healthcare (Jensen et al., 2014), astronomy (Scargle, 1982), or climate science (Schneider, 2001). Much of the methodology for time-series analysis assumes that signals are measured systematically at fixed time intervals. However, much real-world data can be *sporadic* (*i.e.*, the signals are sampled irregularly and not all signals are measured each time). A typical example is patient measurements, which are taken when the patient comes for a visit (*e.g.,* sometimes skipping an appointment) and where not every measurement is taken at every visit. Modeling then becomes challenging as such data violates the main assumptions underlying traditional machine learning methods (such as recurrent neural networks).

Recently, the Neural Ordinary Differential Equation (ODE) model (Chen et al., 2018) opened the way for a novel, continuous representation of neural networks. As time is intrinsically continuous, this framework is particularly attractive for time-series analysis. It opens the perspective of tackling

[*]Both authors contributed equally
[†]Corresponding author

Preprint. Under review.

the issue of irregular sampling in a natural fashion, by integrating the dynamics over whatever time interval needed. Up to now however, such ODE dynamics have been limited to the continuous *generation* of observations (*e.g.*, decoders in variational auto-encoders (VAEs) (Kingma & Welling, 2013) or normalizing flows (Rezende et al., 2014)).

Instead of the encoder-decoder architecture where the ODE part is decoupled from the input processing, we introduce a tight integration by *interleaving* the ODE and the input processing steps. Conceptually, this allows us to drive the dynamics of the ODE directly by the incoming sporadic inputs. To this end, we propose (1) a continuous time version of the Gated Recurrent Unit and (2) a Bayesian update network that processes the sporadic observations. We combine these two ideas to form the GRU-ODE-Bayes method.

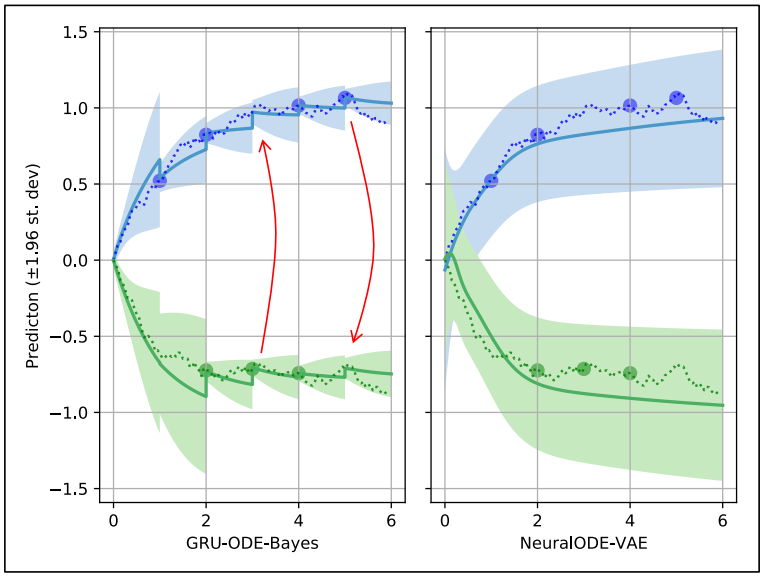


Figure 1: Comparison of GRU-ODE-Bayes and NeuralODE-VAE on a 2D Ornstein-Uhlenbeck process with highly correlated Wiener processes ($\rho = 0.99$). Dots are the values of the actual underlying process (dotted lines) from which the sporadic observations are obtained. Solid lines and shaded areas are the inferred means and 95% confidence intervals. Note the smaller errors and smaller variance of GRU-ODE-Bayes vs. NeuralODE-VAE. Note also that GRU-ODE-Bayes can infer that a jump in one variable also implies a jump in the other unobserved one (red arrows). Similarly, it also learns the reduction of variance resulting from a new incoming observation.

The tight coupling between observation processing and ODE dynamics allows the proposed method to model fine-grained nonlinear dynamical interactions between the variables. As illustrated in Figure 1, GRU-ODE-Bayes can (1) quickly infer the unknown parameters of the underlying stochastic process and (2) learn the correlation between its variables (red arrows in Figure 1). In contrast, the encoder-decoder based method NeuralODE-VAE proposed by Chen et al. (2018) captures the general structure of the process without being able to recover detailed interactions between the variables (see Section 4 for detailed comparison).

Our model enjoys important theoretical properties. We frame our analysis in a general way by considering that observations follow the dynamics driven by a stochastic differential equation (SDE). In Section 4 and Appendix H, we show that GRU-ODE-Bayes can exactly represent the corresponding Fokker-Planck dynamics in the special case of the Ornstein-Uhlenbeck process, as well as in generalized versions of it. We further perform an empirical evaluation and show that our method outperforms the state of the art on healthcare and climate data (Section 5).

1.1 Problem statement

We consider the general problem of forecasting on N sporadically observed D-dimensional time series. For example, data from N patients where D clinical longitudinal variables can potentially be measured. Each time series $i \in \{1, \dots, N\}$ is measured at K_i time points specified by a vector

of observation times $\mathbf{t}_i \in \mathbb{R}^{K_i}$. The values of these observations are specified by a matrix of observations $\mathbf{y}_i \in \mathbb{R}^{K_i \times D}$ and an observation mask $\mathbf{m}_i \in \{0,1\}^{K_i \times D}$ (to indicate which of the variables are measured at each time point).

We assume that observations $\mathbf{y}_i$ are sampled from the realizations of a D-dimensional stochastic process $\mathbf{Y}(t)$ whose dynamics is driven by an unknown SDE:

$$d\mathbf{Y}(t) = \mu(\mathbf{Y}(t))dt + \sigma(\mathbf{Y}(t))d\mathbf{W}(t), \tag{1}$$

where $d\mathbf{W}(t)$ is a Wiener process. The distribution of $\mathbf{Y}(t)$ then evolves according to the celebrated Fokker-Planck equation (Risken, 1996). We refer to the mean and covariance parameters of its probability density function (PDF) as $\mu_{\mathbf{Y}}(t)$ and $\Sigma_{\mathbf{Y}}(t)$.

Our goal will be to model the unknown temporal functions $\mu_{\mathbf{Y}}(t)$ and $\Sigma_{\mathbf{Y}}(t)$ from the sporadic measurements $\mathbf{y}_i$. These are obtained by sampling the random vectors $\mathbf{Y}(t)$ at times $\mathbf{t}_i$ with some observation noise $\boldsymbol{\epsilon}$. Not all dimensions are sampled each time, resulting in missing values in $\mathbf{y}_i$.

This SDE formulation is general. It embodies the natural assumption that seemingly identical processes can evolve differently because of unobserved information. In the case of intensive care, as developed in Section 5, it reflects the evolving uncertainty regarding the patient's future condition.

2 Proposed method

At a high level, we propose a dual mode system consisting of (1) a GRU-inspired continuous-time state evolution (GRU-ODE) that *propagates* in time the hidden state $\mathbf{h}$ of the system between observations and (2) a network that *updates* the current hidden state to incorporate the incoming observations (GRU-Bayes). The system switches from propagation to update and back whenever a new observation becomes available.

We also introduce an observation model $f_{\text{obs}}(\mathbf{h}(t))$ mapping $\mathbf{h}$ to the estimated parameters of the observations distribution $\mu_{\mathbf{Y}}(t)$ and $\Sigma_{\mathbf{Y}}(t)$ (details in Appendix E). GRU-ODE then explicitly learns the Fokker-Planck dynamics of Eq. 1. This procedure allows end-to-end training of the system to minimize the loss with respect to the sporadically sampled observations $\mathbf{y}$.

2.1 GRU-ODE derivation

To derive the GRU-based ODE, we first show that the GRU proposed by Cho et al. (2014) can be written as a difference equation. First, let $\mathbf{r}_t$, $\mathbf{z}_t$, and $\mathbf{g}_t$ be the reset gate, update gate, and update vector of the GRU:

$$\begin{aligned}
\mathbf{r}_t &= \sigma(W_r \mathbf{x}_t + U_r \mathbf{h}_{t-1} + \mathbf{b}_r) \\
\mathbf{z}_t &= \sigma(W_z \mathbf{x}_t + U_z \mathbf{h}_{t-1} + \mathbf{b}_z) \\
\mathbf{g}_t &= \tanh(W_h \mathbf{x}_t + U_h(\mathbf{r}_t \odot \mathbf{h}_{t-1}) + \mathbf{b}_h),
\end{aligned} \tag{2}$$

where $\odot$ is the elementwise product. Then the standard update for the hidden state $\mathbf{h}$ of the GRU is

$$\mathbf{h}_t = \mathbf{z}_t \odot \mathbf{h}_{t-1} + (1 - \mathbf{z}_t) \odot \mathbf{g}_t.$$

We can also write this as $\mathbf{h}_t = \mathbf{GRU}(\mathbf{h}_{t-1}, \mathbf{x}_t)$. By subtracting $\mathbf{h}_{t-1}$ from this state update equation and factoring out $(1 - \mathbf{z}_t)$, we obtain a difference equation

$$\begin{aligned}
\Delta \mathbf{h}_t = \mathbf{h}_t - \mathbf{h}_{t-1} &= \mathbf{z}_t \odot \mathbf{h}_{t-1} + (1 - \mathbf{z}_t) \odot \mathbf{g}_t - \mathbf{h}_{t-1} \\
&= (1 - \mathbf{z}_t) \odot (\mathbf{g}_t - \mathbf{h}_{t-1}).
\end{aligned}$$

This difference equation naturally leads to the following ODE for $\mathbf{h}(t)$:

$$\frac{d\mathbf{h}(t)}{dt} = (1 - \mathbf{z}(t)) \odot (\mathbf{g}(t) - \mathbf{h}(t)), \tag{3}$$

where $\mathbf{z}$, $\mathbf{g}$, $\mathbf{r}$ and $\mathbf{x}$ are the continuous counterpart of Eq. 2. See Appendix A for the explicit form.

We name the resulting system *GRU-ODE*. Similarly, we derive the *minimal GRU-ODE*, a variant based on the minimal GRU (Zhou et al., 2016), described in appendix G.

In case *continuous observations* or *control signals* are available, they can be naturally fed to the GRU-ODE input $\mathbf{x}(t)$. For example, in the case of clinical trials, the administered daily doses of the drug under study can be used to define a continuous input signal. If no continuous input is available, then nothing is fed as $\mathbf{x}(t)$ and the resulting ODE in Eq. 3 is autonomous, with $\mathbf{g}(t)$ and $\mathbf{z}(t)$ only depending on $\mathbf{h}(t)$.

2.2 General properties of GRU-ODE

GRU-ODE enjoys several useful properties:

Boundedness. First, the hidden state $\mathbf{h}(t)$ stays within the $[-1, 1]$ range[3]. This restriction is crucial for the compatibility with the GRU-Bayes model and comes from the negative feedback term in Eq. 3, which stabilizes the resulting system. In detail, if the j-th dimension of the starting state $\mathbf{h}(0)$ is within $[-1, 1]$, then $\mathbf{h}(t)_j$ will always stay within $[-1, 1]$ because

$$\left.\frac{d\mathbf{h}(t)_j}{dt}\right|_{t:\mathbf{h}(t)_j=1} \leq 0 \quad \text{and} \quad \left.\frac{d\mathbf{h}(t)_j}{dt}\right|_{t:\mathbf{h}(t)_j=-1} \geq 0.$$

This can be derived from the ranges of $\mathbf{z}$ and $\mathbf{g}$ in Eq. 2. Moreover, would $\mathbf{h}(0)$ start outside of the $[-1, 1]$ region, the negative feedback would quickly push $\mathbf{h}(t)$ into this region, making the system also robust to numerical errors.

Continuity. Second, GRU-ODE is Lipschitz continuous with constant $K = 2$. Importantly, this means that GRU-ODE encodes a *continuity prior* for the latent process $\mathbf{h}(t)$. This is in line with the assumption of a continuous hidden process generating observations (Eq. 1). In Section 5.5, we demonstrate empirically the importance of this prior in the small-sample regime.

General numerical integration. As a parametrized ODE, GRU-ODE can be integrated with any numerical solver. In particular, adaptive step size solvers can be used. Our model can then afford large time steps when the internal dynamics is slow, taking advantage of the continuous time formulation of Eq. 3. It can also be made faster with sophisticated ODE integration methods. We implemented the following methods: Euler, explicit midpoint, and Dormand-Prince (an adaptive step size method). Appendix C illustrates that the Dormand-Prince method requires fewer time steps.

2.3 GRU-Bayes

GRU-Bayes is the module that processes the sporadically incoming observations to update the hidden vectors, and hence the estimated PDF of $\mathbf{Y}(t)$. This module is based on a standard GRU and thus operates in the region $[-1, 1]$ that is required by GRU-ODE. In particular, GRU-Bayes is able to update $\mathbf{h}(t)$ to any point in this region. Any adaptation is then within reach with a *single observation*.

To feed the GRU unit inside GRU-Bayes with a non-fully-observed vector, we first preprocess it with an observation mask using f_{prep}, as described in Appendix D. For a given time series, the resulting update for its k-th observation $\mathbf{y}[k]$ at time $t = \mathbf{t}[k]$ with mask $\mathbf{m}[k]$ and hidden vector $\mathbf{h}(t_-)$ is

$$\mathbf{h}(t_+) = \mathbf{GRU}(\mathbf{h}(t_-), f_{\text{prep}}(\mathbf{y}[k], \mathbf{m}[k], \mathbf{h}(t_-))), \tag{4}$$

where $\mathbf{h}(t_-)$ and $\mathbf{h}(t_+)$ denote the hidden representation before and after the jump from GRU-Bayes update. We also investigate an alternative option where the $\mathbf{h}(t)$ is updated by each observed dimension *sequentially*. We call this variant *GRU-ODE-Bayes-seq* (see Appendix F for more details).

2.4 GRU-ODE-Bayes

The proposed GRU-ODE-Bayes combines GRU-ODE and GRU-Bayes. The GRU-ODE is used to evolve the hidden state $\mathbf{h}(t)$ in continuous time between the observations and GRU-Bayes transforms the hidden state, based on the observation $\mathbf{y}$, from $\mathbf{h}(t_-)$ to $\mathbf{h}(t_+)$. As best illustrated in Figure 2, the alternation between GRU-ODE and GRU-Bayes results in an ODE with *jumps*, where the jumps are at the locations of the observations.

[3]We use the notation $[-1, 1]$ to also mean multi-dimensional range (*i.e.*, all elements are within $[-1, 1]$).

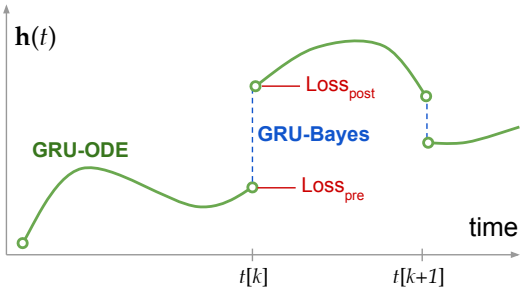


Figure 2: GRU-ODE-Bayes uses GRU-ODE to evolve the hidden state between two observation times $t[k]$ and $t[k+1]$. GRU-Bayes processes the observations and updates the hidden vector $\mathbf{h}$ in a discrete fashion, reflecting the additional information brought in by the observed data.

Objective function

To train the model using sporadically-observed samples, we introduce two losses. The first loss, Loss_{pre}, is computed before the observation update and is the negative log-likelihood (NegLL) of the observations. For the observation of a single sample, we have (for readability we drop the time indexing):

$$\text{Loss}_{\text{pre}} = -\sum_{j=1}^{D} m_j \log p(y_j|\theta = f_{\text{obs}}(\mathbf{h}_-)_j),$$

where m_j is the observation mask and $f_{\text{obs}}(\mathbf{h}_-)_j$ are the parameters of the distribution before the update, for dimension j. Thus, the error is only computed on the observed values of $\mathbf{y}$.

For the second loss, let p_{pre} denote the predicted distributions (from $\mathbf{h}_-$) before GRU-Bayes. With p_{obs}, the PDF of $\mathbf{Y}(t)$ given the noisy observation (with noise vector $\boldsymbol{\epsilon}$), we first compute the analogue of the Bayesian update:

$$p_{\text{Bayes},j} \propto p_{\text{pre},j} \cdot p_{\text{obs},j}.$$

Let p_{post} denote the predicted distribution (from $\mathbf{h}_+$) after applying GRU-Bayes. We then define the post-jump loss as the KL-divergence between p_{Bayes} and p_{post}:

$$\text{Loss}_{\text{post}} = \sum_{j=1}^{D} m_j D_{KL}(p_{\text{Bayes},j}||p_{\text{post},j}).$$

In this way, we force our model to learn to mimic a Bayesian update.

Similarly to the pre-jump loss, $\text{Loss}_{\text{post}}$ is computed only for the observed dimensions. The total loss is then obtained by adding both losses with a weighting parameter λ.

For binomial and Gaussian distributions, computing $\text{Loss}_{\text{post}}$ can be done analytically. In the case of Gaussian distribution we can compute the Bayesian updated mean μ_{Bayes} and variance σ^2_{Bayes} as

$$\mu_{\text{Bayes}} = \frac{\sigma^2_{\text{obs}}}{\sigma^2_{\text{pre}} + \sigma^2_{\text{obs}}}\mu_{\text{pre}} + \frac{\sigma^2_{\text{pre}}}{\sigma^2_{\text{pre}} + \sigma^2_{\text{obs}}}\mu_{\text{obs}}$$

$$\sigma^2_{\text{Bayes}} = \frac{\sigma^2_{\text{pre}} . \sigma^2_{\text{obs}}}{\sigma^2_{\text{pre}} + \sigma^2_{\text{obs}}},$$

where for readability we dropped the dimension sub-index. In many real-world cases, the observation noise $\sigma^2_{\text{obs}} \ll \sigma^2_{\text{pre}}$, in which case p_{Bayes} is just the observation distribution: $\mu_{\text{Bayes}} = \mu_{\text{obs}}$ and $\sigma^2_{\text{Bayes}} = \sigma^2_{\text{obs}}$.

Algorithm 1 GRU-ODE-Bayes

Input: Initial state $\mathbf{h}_0$,
observations $\mathbf{x}$, mask $\mathbf{m}$,
observation times $\mathbf{t}$, final time T.
Initialize time $= 0$, loss $= 0$, $\mathbf{h} = \mathbf{h}_0$.
for $k = 1$ **to** K **do**
{ODE evolution to $\mathbf{t}[k]$}
$\mathbf{h} = \textbf{GRU-ODE}(\mathbf{h}, \text{time}, \mathbf{t}[k])$
time $= t[k]$
{Pre-jump loss}
loss += $\text{Loss}_{\text{pre}}(\mathbf{y}[k], \mathbf{m}[k], \mathbf{h})$
{Update}
$\mathbf{h} = \textbf{GRU-Bayes}(\mathbf{y}[k], \mathbf{m}[k], \mathbf{h})$
{Post-jump loss}
loss += λ. $\text{Loss}_{\text{post}}(\mathbf{y}[k], \mathbf{m}[k], \mathbf{h})$
end for
{ODE evolution to T}
$\mathbf{h} = \textbf{GRU-ODE}(\mathbf{h}, \mathbf{t}[N_K], T)$
return $(\mathbf{h}, \text{loss})$

2.5 Implementation

The pseudocode of GRU-ODE-Bayes is depicted in Algorithm 1, where a forward pass is shown for a single time series [4]. For mini-batching several time series we sort the observation times across all time series and for each unique time point $\mathbf{t}[k]$, we create a list of the time series that have observations. The main loop of the algorithm iterates over this set of unique time points. In the GRU-ODE step, we propagate all hidden states jointly. The GRU-Bayes update and the loss calculation are only executed on the time series that have observation at that particular time point. The complexity of our approach then scales linearly with the number of observations and quadratically with the dimension of the observations. When memory cost is a bottleneck, the gradient can be computed using the adjoint method, without backpropagating through the solver operations (Chen et al., 2018).

3 Related research

Machine learning has a long history in time series modelling (Mitchell, 1999; Gers et al., 2000; Wang et al., 2006; Chung et al., 2014). However, recent massive real-world data collection, such as electronic health records (EHR), increase the need for models capable of handling such complex data (Lee et al., 2017). As stated in the introduction, their *sporadic* nature is the main difficulty.

To address the nonconstant sampling, a popular approach is to recast observations into fixed duration time bins. However, this representation results in missing observation both in time and across features dimensions. This makes the direct usage of neural network architectures tricky. To overcome this issue, the main approach consists in some form of data imputation and jointly feeding the observation mask and times of observations to the recurrent network (Che et al., 2018; Choi et al., 2016a; Lipton et al., 2016; Du et al., 2016; Choi et al., 2016b; Cao et al., 2018). This approach strongly relies on the assumption that the network will learn to process true and imputed samples differently. Despite some promising experimental results, there is no guarantee that it will do so. Some researchers have tried to alleviate this limitation by introducing more meaningful data representation for sporadic time series (Rajkomar et al., 2018; Razavian & Sontag, 2015; Ghassemi et al., 2015).

Others have addressed the missing data problem with generative probabilistic models. Among those, (multitask) Gaussian processes (GP) are the most popular by far (Bonilla et al., 2008). They have been used for smart imputation before a RNN or CNN architecture (Futoma et al., 2017; Moor et al., 2019), for modelling a hidden process in joint models (Soleimani et al., 2018), or to derive informative representations of time series (Ghassemi et al., 2015). GPs have also been used for direct forecasting (Cheng et al., 2017). However, they usually suffer from high uncertainty outside the observation support, are computationally intensive (Quiñonero-Candela & Rasmussen, 2005), and learning the optimal kernel is tricky. Neural Processes, a neural version of GPs, have also been introduced by Garnelo et al. (2018).

Most recently, the seminal work of Chen et al. (2018) suggested a continuous version of neural networks that overcomes the limits imposed by discrete-time recurrent neural networks. Coupled with a variational auto-encoder architecture (Kingma & Welling, 2013), it proposed a natural way of generating irregularly sampled data. However, it transferred the difficult task of processing sporadic data to the encoder, which is a discrete-time RNN. Related auto-encoder approaches with sequential latents operating in discrete time have also been proposed (Krishnan et al., 2015, 2017). These models rely on classical RNN architectures in their inference networks, hence not addressing the sporadic nature of the data. What is more, if they have been shown useful for smoothing and counterfactual inference, their formulation is less suited for forecasting. Our method also has connections to the Extended Kalman Filter (EKF) that models the dynamics of the distribution of processes in continuous time. However, the practical applicability of the EKF is limited because of the linearization of the state update and the difficulties involved in identifying its parameters. Importantly, the ability of the GRU to learn long-term dependencies is a significant advantage.

[4] Code is available in the following anonymous repository : `https://github.com/edebrouwer/gru_ode_bayes`

4 Application to synthetic SDEs

Figure 1 illustrates the capabilities of our approach compared to NeuralODE-VAE on data generated from a process driven by a multivariate Ornstein-Uhlenbeck (OU) SDE with random parameters. Compared to NeuralODE-VAE, which retrieves the average dynamics of the samples, our approach detects the correlation between both features and updates its predictions more finely as new observations arrive. In particular, note that GRU-ODE-Bayes updates its prediction and confidence on a feature even when only the other one is observed, taking advantage from the fact that they are correlated. This can be seen on the left pane of Figure 1 where at time $t = 3$, Dimension 1 (blue) is updated because of the observation of Dimension 2 (green).

By directly feeding sporadic inputs into the ODE, GRU-ODE-Bayes sequentially *filters* the hidden state and thus estimates the PDF of the future observations. This is the core strength of the proposed method, allowing it to perform long-term predictions.

In Appendix H, we further show that our model can exactly represent the dynamics of multivariate OU process with random variables. Our model can also handle nonlinear SDEs as shown in Appendix I where we present an example inspired by the Brusselator (Prigogine, 1982), a chaotic ODE.

5 Empirical evaluation

We evaluated our model on two data sets from different application areas: healthcare and climate forecasting. In both applications, we assume the data consists of noisy observations from an underlying unobserved latent process as in Eq. 1. We focused on the general task of forecasting the time series at future time points. Models are trained to minimize negative log-likelihood.

5.1 Baselines

We used a comprehensive set of state-of-the-art baselines to compare the performance of our method. All models use the same hidden size representation and comparable number of parameters.

NeuralODE-VAE (Chen et al., 2018). We model the time derivative of the hidden representation as a 2-layer MLP. To take missingness across features into account, we add a mechanism to feed an observation mask.

Imputation Methods. We implemented two imputation methods as described in Che et al. (2018): *GRU-Simple* and *GRU-D*.

Sequential VAEs (Krishnan et al., 2015, 2017). We extended the deep Kalman filter architecture by feeding an observation mask and updating the loss function accordingly.

T-LSTM (Baytas et al., 2017). We reused the proposed time-aware LSTM cell to design a forecasting RNN with observation mask.

5.2 Electronic health records

Electronic Health Records (EHR) analysis is crucial to achieve data-driven personalized medicine (Lee et al., 2017; Goldstein et al., 2017; Esteva et al., 2019). However, efficient modeling of this type of data remains challenging. Indeed, it consists of *sporadically* observed longitudinal data with the extra hurdle that there is no standard way to align patients trajectories (*e.g.*, at hospital admission, patients might be in very different state of progression of their condition). Those difficulties make EHR analysis well suited for GRU-ODE-Bayes.

We use the publicly available MIMIC-III clinical database (Johnson et al., 2016), which contains EHR for more than 60,000 critical care patients. We select a subset of 21,250 patients with sufficient observations and extract 96 different longitudinal real-valued measurements over a period of 48 hours after patient admission. We refer the reader to Appendix J for further details on the cohort selection. We focus on the predictions of the next 3 measurements after a 36-hour observation window.

5.3 Climate forecast

From short-term weather forecast to long-range prediction or assessment of systemic changes, such as global warming, climatic data has always been a popular application for time-series analysis. This data is often considered to be regularly sampled over long periods of time, which facilitates their statistical analysis. Yet, this assumption does not usually hold in practice. Missing data are a problem that is repeatedly encountered in climate research because of, among others, measurement errors, sensor failure, or faulty data acquisition. The actual data is then sporadic and researchers usually resort to imputation before statistical analysis (Junninen et al., 2004; Schneider, 2001).

We use the publicly available United State Historical Climatology Network (USHCN) daily data set (Menne et al.), which contains measurements of 5 climate variables (daily temperatures, precipitation, and snow) over 150 years for 1,218 meteorological stations scattered over the United States. We selected a subset of 1,114 stations and an observation window of 4 years (between 1996 and 2000). To make the time series sporadic, we subsample the data such that each station has an average of around 60 observations over those 4 years. Appendix K contains additional details regarding this procedure. The task is then to predict the next 3 measurements after the first 3 years of observation.

5.4 Results

We report the performance using 5-fold cross-validation. Hyperparameters (dropout and weight decay) are chosen using an inner holdout validation set (20%) (More details in Appendix N). Performance metrics for both tasks (NegLL and MSE) are reported in Table 1. GRU-ODE-Bayes handles the sporadic data more naturally and can more finely model the dynamics and correlations between the observed features, which results in higher performance than other methods for both data sets. In particular, GRU-ODE-Bayes unequivocally outperforms all other methods on both data sets.

Table 1: Forecasting results.

	USHCN-DAILY		MIMIC-III	
MODEL	MSE	NEGLL	MSE	NEGLL
NEURALODE-VAE	0.96 ± 0.11	1.46 ± 0.10	0.89 ± 0.01	1.35 ± 0.01
NEURALODE-VAE-MASK	0.83 ± 0.10	1.36 ± 0.05	0.89 ± 0.01	1.36 ± 0.01
SEQUENTIAL VAE	0.83 ± 0.07	1.37 ± 0.06	0.92 ± 0.09	1.39 ± 0.07
GRU-SIMPLE	0.75 ± 0.12	1.23 ± 0.10	0.82 ± 0.05	1.21 ± 0.04
GRU-D	0.53 ± 0.06	0.99 ± 0.07	0.79 ± 0.06	1.16 ± 0.05
T-LSTM	0.59 ± 0.11	1.67 ± 0.50	0.62 ± 0.05	1.02 ± 0.02
GRU-ODE-BAYES	$\mathbf{0.43} \pm 0.07$	$\mathbf{0.84} \pm 0.11$	$\mathbf{0.48} \pm 0.01$	$\mathbf{0.83} \pm 0.04$

5.5 Impact of continuity prior

To illustrate the capabilities of the derived GRU-ODE cell presented in Section 2.1, we consider the case of time series forecasting with low sample size. In the realm of EHR prediction, this could be framed as a *rare disease* setup, where data is available for few patients only. In this case of scarce number of samples, the continuity prior embedded in GRU-ODE is crucially important as it provides important prior information about the underlying process.

To highlight the importance of the GRU-ODE cell, we compare two versions of our model : the classical GRU-ODE-Bayes and one where the GRU-ODE cell is replaced by a discretized autonomous GRU. We call the latter *GRU-Discretized-Bayes*. Table 2 shows the results for MIMIC-III with varying number of patients in the training set. While our discretized version matches the continuous one on the full data set, GRU-ODE cell achieves higher accuracy when the number of samples is low, highlighting the importance of the continuity prior. Log-likelihood results are given in Appendix L.

[†] Statistically not different from best (p-value > 0.6 with paired t-test).

Table 2: Comparison between GRU-ODE and discretized version in the small-sample regime (MSE).

MODEL	1,000 PATIENTS	2,000 PATIENTS	FULL
NEURALODE-VAE-MASK	0.94 ± 0.01	0.94 ± 0.01	0.89 ± 0.01
GRU-DISCRETIZED-BAYES	0.87 ± 0.02	0.77 ± 0.02	$\mathbf{0.46} \pm 0.05$
GRU-ODE-BAYES	$\mathbf{0.77} \pm 0.01$	$\mathbf{0.72} \pm 0.01$	$\mathbf{0.48}^{\dagger} \pm 0.01$

6 Conclusion and future work

We proposed a model combining two novel techniques, GRU-ODE and GRU-Bayes, which allows feeding sporadic observations into a continuous ODE dynamics describing the evolution of the probability distribution of the data. Additionally, we showed that this filtering approach enjoys attractive representation capabilities. Finally, we demonstrated the value of GRU-ODE-Bayes on both synthetic and real-world data. Moreover, while a discretized version of our model performed well on the full MIMIC-III data set, the continuity prior of our ODE formulation proves particularly important in the small-sample regime, which is particularly relevant for real-world clinical data where many data sets remain relatively modest in size.

In this work, we focused on time-series data with Gaussian observations. However, GRU-ODE-Bayes can also be extended to binomial and multinomial observations since the respective NegLL and KL-divergence are analytically tractable. This allows the modeling of sporadic observations of both discrete and continuous variables.

Acknowledgements

Edward De Brouwer is funded by a FWO-SB grant. Yves Moreau is funded by (1) Research Council KU Leuven: C14/18/092 SymBioSys3; CELSA-HIDUCTION, (2) Innovative Medicines Initiative: MELLODY, and (3) Flemish Government (ELIXIR Belgium, IWT, FWO 06260). Computational resources and services used in this work were provided by the VSC (Flemish Supercomputer Center), funded by the Research Foundation - Flanders (FWO) and the Flemish Government – department EWI. We also gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

References

Baytas, I. M., Xiao, C., Zhang, X., Wang, F., Jain, A. K., and Zhou, J. Patient subtyping via time-aware lstm networks. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 65–74. ACM, 2017.

Bonilla, E. V., Chai, K. M., and Williams, C. Multi-task gaussian process prediction. In *Advances in neural information processing systems*, pp. 153–160, 2008.

Cao, W., Wang, D., Li, J., Zhou, H., Li, L., and Li, Y. Brits: Bidirectional recurrent imputation for time series. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 31*, pp. 6776–6786. Curran Associates, Inc., 2018.

Che, Z., Purushotham, S., Cho, K., Sontag, D., and Liu, Y. Recurrent neural networks for multivariate time series with missing values. *Scientific reports*, 8(1):6085, 2018.

Chen, T. Q., Rubanova, Y., Bettencourt, J., and Duvenaud, D. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems, 2018*, 2018.

Cheng, L.-F., Darnell, G., Chivers, C., Draugelis, M. E., Li, K., and Engelhardt, B. E. Sparse multi-output gaussian processes for medical time series prediction. *arXiv preprint arXiv:1703.09112*, 2017.

Cho, K., van Merrienboer, B., Gülçehre, Ç., Bougares, F., Schwenk, H., and Bengio, Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078, 2014. URL `http://arxiv.org/abs/1406.1078`.

Choi, E., Bahadori, M. T., Schuetz, A., Stewart, W. F., and Sun, J. Doctor ai: Predicting clinical events via recurrent neural networks. In *Machine Learning for Healthcare Conference*, pp. 301–318, 2016a.

Choi, E., Bahadori, M. T., Sun, J., Kulas, J., Schuetz, A., and Stewart, W. Retain: An interpretable predictive model for healthcare using reverse time attention mechanism. In *Advances in Neural Information Processing Systems*, pp. 3504–3512, 2016b.

Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

Du, N., Dai, H., Trivedi, R., Upadhyay, U., Gomez-Rodriguez, M., and Song, L. Recurrent marked temporal point processes: Embedding event history to vector. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1555–1564. ACM, 2016.

Esteva, A., Robicquet, A., Ramsundar, B., Kuleshov, V., DePristo, M., Chou, K., Cui, C., Corrado, G., Thrun, S., and Dean, J. A guide to deep learning in healthcare. *Nature medicine*, 25(1):24, 2019.

Futoma, J., Hariharan, S., and Heller, K. Learning to detect sepsis with a multitask gaussian process rnn classifier. *arXiv preprint arXiv:1706.04152*, 2017.

Garnelo, M., Schwarz, J., Rosenbaum, D., Viola, F., Rezende, D. J., Eslami, S., and Teh, Y. W. Neural processes. *arXiv preprint arXiv:1807.01622*, 2018.

Gers, F. A., Schmidhuber, J., and Cummins, F. Learning to forget: Continual prediction with lstm. *Neural Computation*, 12(10):2451–2471, 2000.

Ghassemi, M., Pimentel, M. A., Naumann, T., Brennan, T., Clifton, D. A., Szolovits, P., and Feng, M. A multivariate timeseries modeling approach to severity of illness assessment and forecasting in icu with sparse, heterogeneous clinical data. In *AAAI*, pp. 446–453, 2015.

Goldstein, B. A., Navar, A. M., Pencina, M. J., and Ioannidis, J. Opportunities and challenges in developing risk prediction models with electronic health records data: a systematic review. *Journal of the American Medical Informatics Association*, 24(1):198–208, 2017.

Jensen, A. B., Moseley, P. L., Oprea, T. I., Ellesøe, S. G., Eriksson, R., Schmock, H., Jensen, P. B., Jensen, L. J., and Brunak, S. Temporal disease trajectories condensed from population-wide registry data covering 6.2 million patients. *Nature communications*, 5:4022, 2014.

Johnson, A. E., Pollard, T. J., Shen, L., Li-wei, H. L., Feng, M., Ghassemi, M., Moody, B., Szolovits, P., Celi, L. A., and Mark, R. G. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3:160035, 2016.

Junninen, H., Niska, H., Tuppurainen, K., Ruuskanen, J., and Kolehmainen, M. Methods for imputation of missing values in air quality data sets. *Atmospheric Environment*, 38(18):2895–2907, 2004.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Krishnan, R. G., Shalit, U., and Sontag, D. Deep kalman filters. *arXiv preprint arXiv:1511.05121*, 2015.

Krishnan, R. G., Shalit, U., and Sontag, D. Structured inference networks for nonlinear state space models. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

Lee, C., Luo, Z., Ngiam, K. Y., Zhang, M., Zheng, K., Chen, G., Ooi, B. C., and Yip, W. L. J. Big healthcare data analytics: Challenges and applications. In *Handbook of Large-Scale Distributed Computing in Smart Healthcare*, pp. 11–41. Springer, 2017.

Lipton, Z. C., Kale, D., and Wetzel, R. Directly modeling missing data in sequences with rnns: Improved classification of clinical time series. In *Machine Learning for Healthcare Conference*, pp. 253–270, 2016.

Menne, M., Williams Jr, C., and Vose, R. Long-term daily climate records from stations across the contiguous united states.

Mitchell, T. M. Machine learning and data mining. *Communications of the ACM*, 42(11):30–36, 1999.

Moor, M., Horn, M., Rieck, B., Roqueiro, D., and Borgwardt, K. Temporal convolutional networks and dynamic time warping can drastically improve the early prediction of sepsis. *arXiv preprint arXiv:1902.01659*, 2019.

Prigogine, I. *From being to becoming*. Freeman, 1982.

Quiñonero-Candela, J. and Rasmussen, C. E. A unifying view of sparse approximate gaussian process regression. *Journal of Machine Learning Research*, 6(Dec):1939–1959, 2005.

Rajkomar, A., Oren, E., Chen, K., Dai, A. M., Hajaj, N., Hardt, M., Liu, P. J., Liu, X., Marcus, J., Sun, M., et al. Scalable and accurate deep learning with electronic health records. *npj Digital Medicine*, 1(1):18, 2018.

Razavian, N. and Sontag, D. Temporal convolutional neural networks for diagnosis from lab tests. *arXiv preprint arXiv:1511.07938*, 2015.

Rezende, D. J., Mohamed, S., and Wierstra, D. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, pp. 1278–1286, 2014.

Risken, H. Fokker-planck equation. In *The Fokker-Planck Equation*, pp. 63–95. Springer, 1996.

Scargle, J. D. Studies in astronomical time series analysis. ii-statistical aspects of spectral analysis of unevenly spaced data. *The Astrophysical Journal*, 263:835–853, 1982.

Schneider, T. Analysis of incomplete climate data: Estimation of mean values and covariance matrices and imputation of missing values. *Journal of climate*, 14(5):853–871, 2001.

Soleimani, H., Hensman, J., and Saria, S. Scalable joint models for reliable uncertainty-aware event prediction. *IEEE transactions on pattern analysis and machine intelligence*, 40(8):1948–1963, 2018.

Wang, J., Hertzmann, A., and Fleet, D. J. Gaussian process dynamical models. In *Advances in neural information processing systems*, pp. 1441–1448, 2006.

Zhou, G.-B., Wu, J., Zhang, C.-L., and Zhou, Z.-H. Minimal gated unit for recurrent neural networks. *International Journal of Automation and Computing*, 13(3):226–234, 2016.

A Full formulation of the GRU-ODE cell

The full ODE equation for GRU-ODE is the following:

$$\frac{d\mathbf{h}(t)}{dt} = (1 - \mathbf{z}(t)) \odot (\mathbf{g}(t) - \mathbf{h}(t)),$$

with

$$\begin{aligned}
\mathbf{r}(t) &= \sigma(W_r \mathbf{x}(t) + U_r \mathbf{h}(t) + \mathbf{b}_r) \\
\mathbf{z}(t) &= \sigma(W_z \mathbf{x}(t) + U_z \mathbf{h}(t) + \mathbf{b}_z) \\
\mathbf{g}(t) &= \tanh(W_h \mathbf{x}(t) + U_h(\mathbf{r}(t) \odot \mathbf{h}(t)) + \mathbf{b}_h).
\end{aligned}$$

Matrices $W \in \mathbb{R}^{H \times D}$, $B \in \mathbb{R}^{H \times H}$ and bias vectors $\mathbf{b} \in \mathbb{R}^H$ are the parameters of the cell. H and D are the dimension of the hidden process and of the inputs respectively.

B Lipschitz continuity of GRU-ODE

As $\mathbf{h}$ is differentiable and continous on t, we know from the mean value theorem that for any $t_a, t_b \in t$, there exists $t^* \in (t_a, t_b)$ such that

$$\mathbf{h}(t_b) - \mathbf{h}(t_a) = \frac{d\mathbf{h}}{dt} \mid_{t^*} (t_b - t_a).$$

Taking the euclidean norm of the previous expression, we find

$$\mid \mathbf{h}(t_b) - \mathbf{h}(t_a) \mid = \mid \frac{d\mathbf{h}}{dt} \mid_{t^*} (t_b - t_a) \mid .$$

Furthermore, we showed that $\mathbf{h}$ is bounded on $[-1, 1]$. Hence, because of the bounded functions appearing in the ODE (sigmoids and hyperbolic tangents), the derivative of $\mathbf{h}$ is itself bounded by $[-2, 2]$. We conclude that $\mathbf{h}(t)$ is Lipschitz continuous with constant $K = 2$.

C Comparison of numerical integration methods

We implemented three numerical integration methods, among which the classical Euler method and the Dormand-Prince method (DOPRI). DOPRI is a popular adaptive step size numerical integration method relying on 2 Runge-Kutta solvers of order 4 and 5. The advantage of adaptive step size methods is that they can tune automatically the number of steps to integrate the ODE until the desired point.

Figure 3 illustrates the number of steps taken by both solvers when given the same data and same ODE. We observe that using an adaptive step size results in half as many time steps. More steps are taken near the observations and as the underlying process becomes smoother, the step size increase, as observed on the right side of the figure. However, each time step requires significantly fewer computations for Euler than for DOPRI, so that Euler's method appears more than competitive on the data and simulations we have considered so far. Nevertheless, DOPRI might still be preferred as default method because of its better numerical stability.

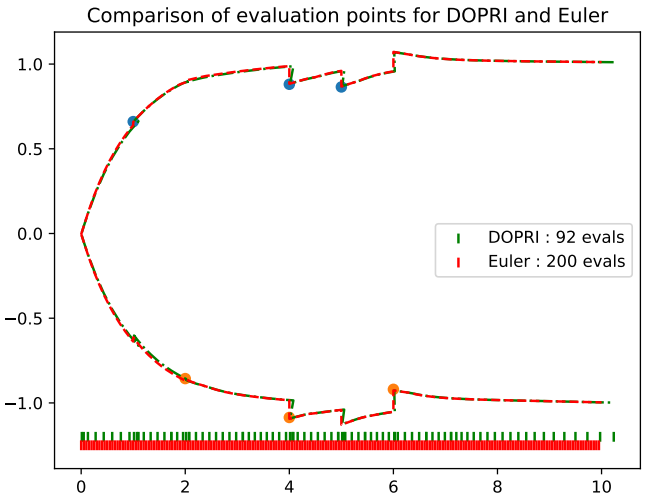


Figure 3: Comparison of Euler and DOPRI numerical integration methods for same inputs and same ODE. Colored ticks on the x axis represent the evaluation time for each method. Dotted lines show the evolution of the estimated mean distribution of the observations while the dots stand for the observations fed to the model.

D Mapping to deal with missingness across features

The preprocessing step f_{prep} for GRU-Bayes takes in the hidden state $\mathbf{h}$ and computes the parameters for the observation PDFs $\theta = f_{\text{obs}}(\mathbf{h}(t))$. In the case of a Gaussian, θ_d contains the means and log-variances for dimension d of $\mathbf{Y}(t)$. Then, we create a vector $\mathbf{q}_d$ that concatenates θ_d with the observed value $\mathbf{y}[k]_d$ and the normalized error term, which for the Gaussian case is $(\mathbf{y}[k]_d - \mu_d)/\sigma_d$, where μ_d and σ_d are the mean and standard deviation derived from θ_d. We then multiply the vectors $\mathbf{q}_d$ by a dimension-specific weight matrix $\mathbf{W}_d$ and apply a ReLU nonlinearity. Next, we zero all results that did not have an observation (by multiplying them with mask m_d). Finally, the concatenation of the results is fed into the GRU unit of GRU-Bayes.

E Observation model mapping

The mapping from hidden $\mathbf{h}$ to the parameters of the distribution $\mu_{\mathbf{Y}(t)}$ and $\log(\Sigma_{\mathbf{Y}(t)})$. For this purpose we use a classical multi-layer perceptron architecture with a 25 dimensional hidden layer. Note that me map to the log of the variance in order to keep it positive.

F GRU-ODE-Bayes-seq

On top of the architecture described in the main bulk of this paper, we also propose a variant which process the sporadic inputs *sequentially*. In other words, GRU-Bayes will update its prediction on the hidden $\mathbf{h}$ for one input dimension after the other rather than jointly. We call this approach GRU-ODE-Bayes-seq.

In this sequential approach for GRU-Bayes, we process one-by-one all dimensions of $\mathbf{y}[k]$ that were *observed* at time $t[k]$ by first applying the preprocessing to each and then sending them to the GRU unit. The preprocessing steps are the same as in the nonsequential scheme (Appendix D) but without concatenation at the end because only one dimension is processed at a time. Note that the preprocessing of dimensions cannot be done in parallel as the hidden state $\mathbf{h}$ changes after each dimension is processed, which affects the computed θ_d and thus the resulting vector $\mathbf{q}_d$.

G Minimal GRU-ODE

Following the same reasoning as for the full GRU cell, we also derived the minimal GRU-ODE cell, based on the minimal GRU cell. The minimal GRU writes :

$$\begin{aligned}\mathbf{f}_t &= \sigma(W_f \mathbf{x}_t + U_f \mathbf{h}_{t-1} + \mathbf{b}_f) \\ \mathbf{h}_t &= \mathbf{f}_t \odot \mathbf{h}_{t-1} + (1 - \mathbf{f}_t) \odot \sigma(W_h \mathbf{x}_t + U_h(\mathbf{h}_{t-1} \odot \mathbf{f}_t) + \mathbf{b}_h)\end{aligned}$$

This can be rewritten as the following difference equation :

$$\Delta \mathbf{h}_t = (1 - \mathbf{f}_t) \odot (\sigma(W_h \mathbf{x}_t + U_h(\mathbf{h}_{t-1} \odot \mathbf{f}_t) + \mathbf{b}_h) - \mathbf{h}_{t-1})$$

Which leads to the corresponding ODE :

$$\frac{d\mathbf{h}(t)}{dt} = (1 - \mathbf{f}(t)) \odot (\sigma(W_h \mathbf{x}(t) + U_h(\mathbf{h}(t) \odot \mathbf{f}(t)) + \mathbf{b}_h) - \mathbf{h}(t))$$

H Application to Ornstein-Uhlenbeck SDEs

We demonstrate the capabilities of our approach on data generated from a process driven by an SDE as in Eq. 1. In particular, we focus on extensions of the multidimensional correlated Ornstein-Uhlenbeck (OU) process with varying parameters. For a particular sample i, the dynamics is given by the following SDE:

$$d\mathbf{Y}_i(t) = \theta_i(\mathbf{r}_i - \mathbf{Y}_i(t))dt + \sigma_i d\mathbf{W}(t), \tag{5}$$

where $\mathbf{W}(t)$ is a D-dimensional correlated Wiener process, $\mathbf{r}_i$ is the vector of targets, and θ_i is the reverting strength constant. For simplicity, we consider θ_i and σ_i parameters as scalars. Each sample $\mathbf{y}_i$ is then obtained via the realization of process (5) with sample-specific parameters.

H.1 Representation capabilities

We now show that our model exactly captures the dynamics of the distribution of $\mathbf{Y}(t)$ as defined in Eq. 5. The evolution of the PDF of a diffusion process is given by the corresponding Fokker-Planck equation. For the OU process, this PDF is Gaussian with time-dependent mean and covariance. Conditioned on a previous observation at time t^*, this gives

$$\begin{aligned}\mathbf{Y}_i(t) \mid \mathbf{Y}_i(t^*) &\sim \mathcal{N}(\mu_{\mathbf{Y}}(t, t^*), \sigma^2_{\mathbf{Y}}(t, t^*)), \\ \mu_{\mathbf{Y}}(t, t^*) &= \mathbf{r}_i + (\mathbf{Y}_i(t^*) - \mathbf{r}_i) \exp(-\theta_i(t - t^*)), \\ \sigma^2_{\mathbf{Y}}(t, t^*) &= \frac{\sigma_i^2}{2\theta_i}(1 - \exp(-2\theta_i(t - t^*))).\end{aligned}$$

Correlation of $\mathbf{Y}(t)$ is constant and equal to ρ, the correlation of the Wiener processes. The dynamics of the mean and variance parameters can be better expressed in the following ODE form:

$$\begin{aligned}\frac{d\mu_{\mathbf{Y}}(t, t^*)}{dt} &= -\theta_i(\mu_{\mathbf{Y}}(t, t^*) - \mathbf{r}_i) \\ \frac{do^2_{\mathbf{Y}}(t, t^*)}{dt} &= -2\theta_i \left(\sigma^2_{\mathbf{Y}}(t, t^*) - \frac{\sigma_i^2}{2\theta_i}\right)\end{aligned} \tag{6}$$

With initial conditions $\mu_{\mathbf{Y}}(0, t^*) = \mathbf{Y}(t^*)$ and $\sigma^2_{\mathbf{Y}}(0, t^*) = 0$. We next investigate how specific versions of this ODE can be represented by our GRU-ODE-Bayes.

H.1.1 Standard Ornstein-Uhlenbeck process

In standard OU, the parameters $\mathbf{r}_i$, σ_i, and θ_i are fixed and identical for all samples. The ODE (6) is linear and can then be represented directly with GRU-ODE by storing $\mu_{\mathbf{Y}}(t, t^*)$ and $\sigma^2_{\mathbf{Y}}(t, t^*)$ in the hidden state $\mathbf{h}(t)$ and matching the Equations (3) and (6). The OU parameters $\mathbf{r}_i$, σ_i and θ_i are learned and encoded in the weights of GRU-ODE. GRU-Bayes then updates the hidden state and stores $\mu_{\mathbf{Y}}(t, t^*)$ and $\sigma^2_{\mathbf{Y}}(t, t^*)$.

Table 3: NegLL and MSE results for 2-dimensional general Ornstein-Uhlenbeck process.

	NEGATIVE LOG-LIKELIHOOD			MSE		
MODEL	RANDOM r	RANDOM LAG	$\rho = 0$	RANDOM R	RANDOM LAG	$\rho = 0$
NEURALODE-VAE-MASK	0.222	0.223	0.204	0.081	0.069	0.081
NEURALODE-VAE	0.183	0.230	0.201	0.085	0.119	0.113
GRU-ODE-BAYES	-1.260	-1.234	-1.187	0.005	0.005	0.006
GRU-ODE-BAYES-MINIMAL	-1.257	-1.226	-1.188	0.005	0.006	0.006
GRU-ODE-BAYES-SEQ	-1.266	-1.225	-1.191	0.005	0.005	0.006
GRU-ODE-BAYES-SEQ-MINIMAL	-1.260	-1.225	-1.189	0.005	0.005	0.006

H.1.2 Generalized Ornstein-Uhlenbeck processes

When parameters are allowed to vary over samples, these have to be encoded in the hidden state of GRU-ODE-Bayes, rather than in the fixed weights. For $\mathbf{r}_i$ and σ_i, GRU-Bayes computes and stores their current estimates as the observations arrive. This is based on previous hidden and current observation as in Eq. 4. The GRU-ODE module then simply has to keep these estimates unchanged between observations:

$$\frac{d\mathbf{r}_i(t)}{dt} = \frac{d\sigma_i(t)}{dt} = 0.$$

This can be easily done by switching off the update gate (*i.e.*, setting $\mathbf{z}(t)$ to 1 for these dimensions). These hidden states can then be used to output the mean and variance in Eq. 6, thus enabling the model to represent generalized Ornstein-Uhlenbeck processes with sample-dependent $\mathbf{r}_i$ and σ_i.

Perfect representation for sample dependent θ_i requires the multiplication of inputs in Eq. 6, which GRU-ODE is not able to perform exactly but should be able to approximate reasonably well. If an exact representation is required, the addition of a bilinear layer is sufficient.

Furthermore, the same reasoning applies when parameters are also allowed to change over time within the same sample. GRU-Bayes is again able to update the hidden vector with the new estimates.

H.1.3 Non-aligned time series

Our approach can also handle samples that would be dephased in time (*i.e*, the observation windows are not aligned on an intrinsic time scale). Longitudinal patient data recorded at different stages of the disease for each patient is one example, developed in Section 5. This setting is naturally handled by the GRU-Bayes module.

H.2 Case Study: 2D Ornstein-Uhlenbeck Process

H.2.1 Setup

We evaluate our model on a 2-dimensional OU process with correlated Brownian motion as defined in Eq. 5. For best illustration of its capabilities, we consider the three following cases.

In the first setting, $\mathbf{r}_i$ varies across samples as $\mathbf{r}_i^1 \sim \mathcal{U}(0.5, 1.5)$ and $\mathbf{r}_i^2 \sim \mathcal{U}(-1.5, -0.5)$. The correlation between the Wiener processes ρ is set to 0.99. We also set $\sigma = 0.1$ and $\theta = 1$. The second case, which we call *random lag* is similar to the first one but adds an extra uniformly distributed random lag to each sample. Samples are then time shifted by some $\Delta_T \sim \mathcal{U}(0, 0.5)$. The third setting is identical to the first but with $\rho = 0$ (*i.e.*, both dimensions are independent and no information is shared between them).

We evaluate all methods and settings on the forecast of samples after time $t = 4$. The training set contains 10,000 samples with an average of 20 observations scattered over a 10-second time interval. Models are trained with a negative log-likelihood objective function, but mean square errors (MSE) are also reported. We compare our methods to NeuralODE-VAE (Chen et al., 2018). Additionally, we consider an extended version of this model where we also feed the observation mask, called NeuralODE-VAE-Mask.

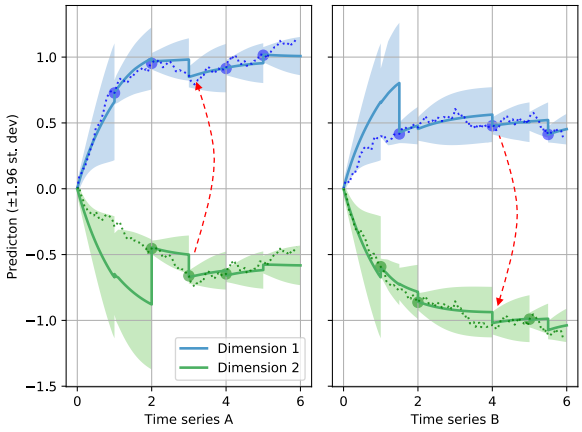


Figure 4: Example of predictions (with shaded confidence intervals) given by GRU-ODE-Bayes for two samples of a correlated 2-dimensional stochastic process (dotted line) with unknown parameters. Dots show the observations. Only a few observations are required for the model to infer the parameters. Additionally, GRU-ODE-Bayes learns the correlation between the dimensions resulting in updates of nonobserved variables (red dashed arrow).

H.2.2 Empirical evaluation

Figure 1 shows a comparison of predictions between NeuralODE-VAE and GRU-ODE-Bayes for the same sample issued from the random $\mathbf{r}_i$ setting. Compared to NeuralODE-VAE, which retrieves the average dynamics of the sample, our approach detects the correlation between both features and updates its predictions more finely as the observations arrive. In particular, note that GRU-ODE-Bayes updates its prediction and confidence on a feature even when only the other one is observed, taking advantage from the fact that they are correlated. This can be seen on the left pane of Figure 1 where at time $t = 3$ Dimension 1 (blue) is updated because of the observation of Dimension 2 (green).

By directly feeding sporadic inputs into the ODE, GRU-ODE-Bayes sequentially *filters* the hidden state and thus estimates the PDF of the future observations. This is the core strength of the proposed method, allowing it to perform long-term predictions. In contrast, NeuralODE-VAE first stores the whole dynamics in a single vector and later maps it to the dynamics of the time series (illustrated in Figure 1).

This analysis is confirmed by the performance results presented in Table 3. Our approach performs better on all setups for both NegLL and MSE. What is more, the method deals correctly with *lags* (*i.e.*, the second setup) as it results in only marginal degradation of NegLL and MSE. When there is no correlation between both dimensions (*i.e.,* $\rho = 0$), the observation of one dimension contains no information on the other and this results in lower performance.

Figure 5 illustrates how GRU-ODE-Bayes updates its prediction and confidence as more and more observations are processed. This example is for the first setup (randomized $\mathbf{r}_i$). Initially, the predictions have large confidence intervals and reflect the general statistics of the training data. Then, observations gradually reduce the variance estimate as the model refines its predictions of the parameter $\mathbf{r}_i$. As more data is processed, the predictions converge to the asymptotic distribution of the underlying process.

I Application to synthetic nonlinear SDE: the Brusselator

On top of the extended multivariate OU process, we also studied a nonlinear SDE. We derived it from the Brusselator ODE, which was proposed by Ilya Prigogine to model autocatalytic reactions (Prigogine, 1982). It is a 2-dimensional process characterized by the following equations:

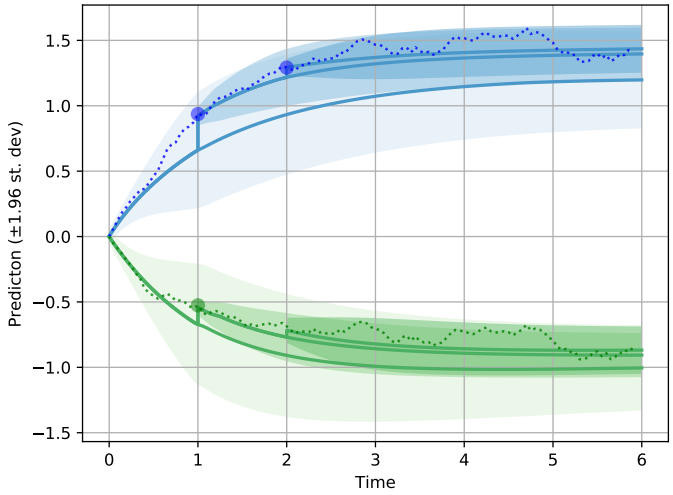


Figure 5: GRU-ODE-Bayes updating its prediction trajectory with every new observation for the random $\mathbf{r}_i$ setup. Shaded regions are propagated confidence intervals conditioned on previous observations.

$$\frac{dx}{dt} = 1 + (b+1)x + ax^2 y$$
$$\frac{dy}{dt} = bx - ax^2 y$$

Where x and y stand for the two dimensions of the process and a and b are parameters of the ODE. This system becomes unstable when $b > 1 + a$. We add a stochastic component to this process to make it the following SDE, which we will model:

$$\begin{aligned} \frac{dx}{dt} &= 1 + (b+1)x + ax^2 y + \sigma dW_1(t) \\ \frac{dy}{dt} &= bx - ax^2 y + \sigma dW_2(t) \end{aligned} \tag{7}$$

Where $dW_1(t)$ and $dW_2(t)$ are correlated Brownian motions with correlation coefficient ρ. We simulate 1,000 trajectories driven by the dynamics given in Eq. 7 with parameters $a = 0.3$ and $b = 1.4$ such that the ODE is unstable. Figure 6 show some realization of this process. The data set we use for training consists in random samples from those trajectories of length 50. We sample sporadically with an average rate of 4 samples every 10 seconds.

Figures 7 show the predictions of the trained model on different samples of the proposed stochastic Brusselator process (newly generated samples). At each point in time are displayed the means and the standard deviation of the *filtered* process. We stress that it means that those predictions only use the observations prior to them. Red arrows show that information is shared between both dimensions of the process. The model is able to pick up the correlation between dimensions to update its belief about one dimension when only the other is observed. The model presented in these figures used 50 dimensional latents with DOPRI solver.

J MIMIC-III: preprocessing details

MIMIC-III is a publicly available database containing deidentified health-related data associated for about 60,000 admissions of patients who stayed in critical care units of the Beth Israel Deaconess

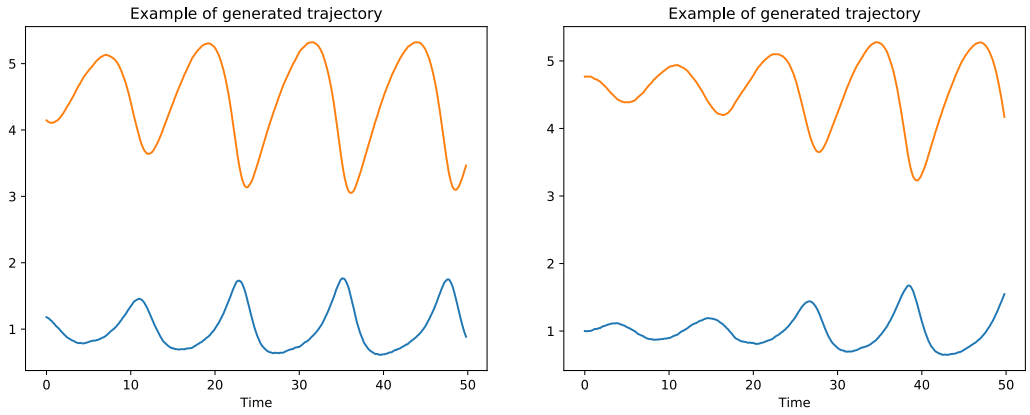


Figure 6: Examples of generated trajectories for the stochastic Brusselator Eq. 7 over 50 seconds. Trajectories vary due to stochastic component and sensitivity to initial conditions. Orange and blue lines represent both components of the process.

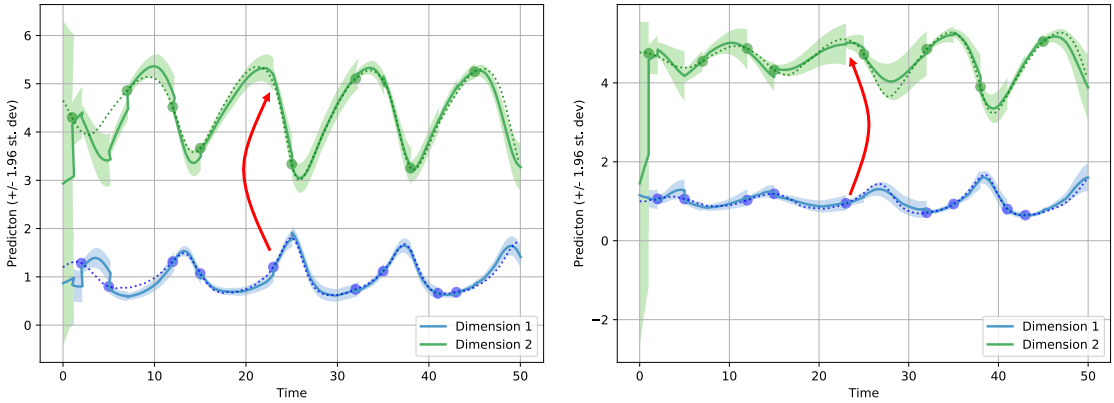


Figure 7: Examples of predicted trajectories for the Brusselator. The model has been trained with DOPRI solver. Solid line shows the predicted *filtered* mean, the shaded areas show the 95% confidence interval while dotted lines represent the true generative process. The dots show the available observations for the filtering. Red arrows show the collapse of the belief function from one dimension to another.

Medical Center between 2001 and 2012. To use the database, researchers must formally request access to the data via `http://mimic.physionet.org`.

J.1 Admission/Patient clean-up

We only take a subset of admissions for our analysis. We select them on the following criteria:

- Keep only patient who are in the metavision system.
- Keep only patients with single admission.
- Keep only patients whose admission is longer than 48 hours, but less than 30 days.
- Remove patients younger than 15 years old at admission time.
- Remove patients without chart events data.
- Remove patients with fewer than 50 measurements over the 48 hours. (This corresponds to measuring only half of retained variable a single time in 48 hours.)

This process restricts the data set to 21,250 patients.

J.2 Variables preprocessing

The subset of 96 variables that we use in our study are shown in Table 4. For each of those, we harmonize the units and drop the uncertain occurrences. We also remove outliers by discarding the measurements outside the 5 standard deviations interval. For models requiring binning of the time series, we map the measurements in 30-minute time bins, which gives 97 bins for 48 hours. When two observations fall in the same bin, they are either averaged or summed depending on the nature of the observation. Using the same taxonomy as in Table 4, lab measurements are averaged, while inputs, outputs, and prescriptions are summed.

This gives a total of 3,082,224 unique measurements across all patients, or an average of 145 measurements per patient over 48 hours.

Retained Features			
Lab measurements	Inputs	Outputs	Prescriptions
Anion Gap	Potassium Chloride	Stool Out Stool	D5W
Bicarbonate	Calcium Gluconate	Urine Out Incontinent	Docusate Sodium
Calcium, Total	Insulin - Regular	Ultrafiltrate Ultrafiltrate	Magnesium Sulfate
Chloride	Heparin Sodium	Gastric Gastric Tube	Potassium Chloride
Glucose	K Phos	Foley	Bisacodyl
Magnesium	Sterile Water	Void	Humulin-R Insulin
Phosphate	Gastric Meds	TF Residual	Aspirin
Potassium	GT Flush	Pre-Admission	Sodium Chloride 0.9% Flush
Sodium	LR	Chest Tube 1	Metoprolol Tartrate
Alkaline Phosphatase	Furosemide (Lasix)	OR EBL	
Asparate Aminotransferase	Solution	Chest Tube 2	
Bilirubin, Total	Hydralazine	Fecal Bag	
Urea Nitrogen	Midazolam (Versed)	Jackson Pratt 1	
Basophils	Lorazepam (Ativan)	Condom Cath	
Eosinophils	PO Intake		
Hematocrit	Insulin - Humalog		
Hemoglobin	OR Crystalloid Intake		
Lymphocytes	Morphine Sulfate		
MCH	D5 1/2NS		
MCHC	Insulin - Glargine		
MCV	Metoprolol		
Monocytes	OR Cell Saver Intake		
Neutrophils	Dextrose 5%		
Platelet Count	Norepinephrine		
RDW	Piggyback		
Red Blood Cells	Packed Red Blood Cells		
White Blood Cells	Phenylephrine		
PTT	Albumin 5%		
Base Excess	Nitroglycerin		
Calculated Total CO2	KCL (Bolus)		
Lactate	Magnesium Sulfate (Bolus)		
pCO2			
pH			
pO2			
PT			
Alanine Aminotransferase			
Albumin			
Specific Gravity			

Table 4: Retained longitudinal features in the intensive care case study.

K USHCN-Daily: preprocessing details

The United States Historical Climatology Network (USHCN) data set contains data from 1,218 centers scattered across the US. The data is publicly available and can be downloaded at the following address: `https://cdiac.ess-dive.lbl.gov/ftp/ushcn_daily/`. All states files contain daily measurements for 5 variables: precipitation, snowfall, snow depth, maximum temperature and minimum temperature.

K.1 Cleaning and subsampling

We first remove all observations with a bad quality flag, then remove all centers that do not have observation before 1970 and after 2001. We then only keep the observations between 1950 and 2000. We subsample the remaining observations to keep on average 5% of the observations of each center. Lastly, we select the last 4 years of the kept series to be used in the analysis.

This process leads to a data set with 1,114 centers, and a total of 386,068 unique observations (or an average of 346 observations per center, sporadically spread over 4 years).

L Small-sample regime: additional results

In the main text of the paper, we presented the results for the Mean Square Error (MSE) for the different data subsets of MIMIC. In Table 5, we present the negative log-likelihood results. They further illustrate that the continuity prior embedded in our GRU-ODE-Bayes strongly helps in the small-sample regime.

Table 5: Vitals forecasting results on MIMIC-III (NegLL) - Low number of samples setting

	1,000 PATIENTS	2,000 PATIENTS	FULL
MODEL	NEGLL	NEGLL	NEGLL
NEURAL-ODE	1.40 ± 0.01	1.39 ± 0.005	1.35 ± 0.01
GRU-DISC-BAYES	1.35 ± 0.01	1.20 ± 0.015	$\mathbf{0.74} \pm 0.04$
GRU-ODE-BAYES	$\mathbf{1.23} \pm 0.006$	$\mathbf{1.13} \pm 0.01$	0.83 ± 0.04

M Computing Infrastructure

All models were run using a NVIDIA P100 GPU with 16GB RAM and 9 CPU cores (Intel(R) Xeon(R) Gold 6140). Implementation was done in Python, using Pytorch as autodifferentitation package. Required packages are available in the code

N Hyper-parameters used

All methods were trained using the same dimension for the hidden $\mathbf{h}$, for sake of fairness. We tuned the following hyper-parameters using a 20% left out validation set:

Dropout rate of 0, 0.1, 0.2 and 0.3.

Weight decay: 0.1, 0.03, 0.01, 0.003, 0.001, 0.0001 and 0.

Learning rate : 0.001 and 0.0001

Best model was selected using early stopping and performance were assessed by applying the best model on a held out test set (10% of the total data). We performed 5-fold cross validation and present the test performance average and standard deviation in all tables.