


Reconstructing in-depth activity for chaotic 3D spatiotemporal excitable media models based on surface data


Cite as: Chaos **33**, 013134 (2023); <https://doi.org/10.1063/5.0126824>

Submitted: 17 September 2022 • Accepted: 22 December 2022 • Published Online: 23 January 2023

 R. Stenger,  S. Herzog,  I. Kottlarz, et al.

COLLECTIONS

 This paper was selected as Featured

 This paper was selected as Scilight



View Online



Export Citation



CrossMark

ARTICLES YOU MAY BE INTERESTED IN

[Jewels from chaos: A fascinating journey from abstract forms to physical objects](#)

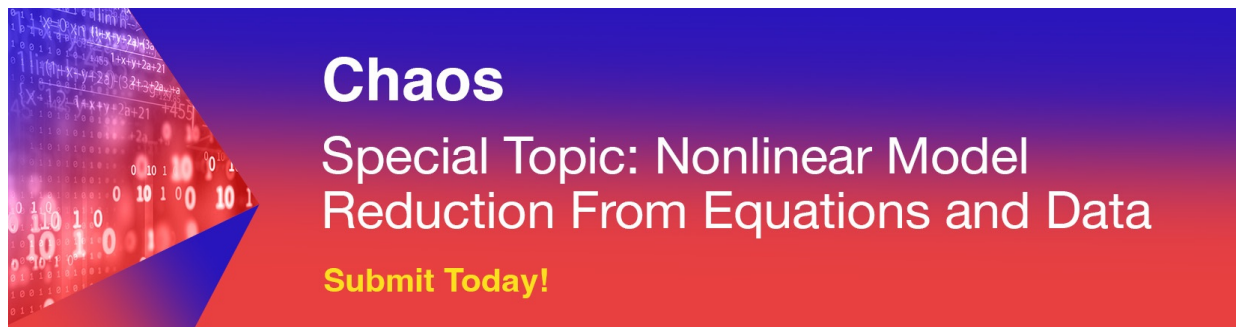
Chaos: An Interdisciplinary Journal of Nonlinear Science **33**, 013132 (2023); <https://doi.org/10.1063/5.0130029>

[A novel method to measure static and dynamic complexity of time series based on visualization curves](#)

Chaos: An Interdisciplinary Journal of Nonlinear Science **33**, 013135 (2023); <https://doi.org/10.1063/5.0119415>

[Symmetry-breaking rhythms in coupled, identical fast-slow oscillators](#)

Chaos: An Interdisciplinary Journal of Nonlinear Science **33**, 011102 (2023); <https://doi.org/10.1063/5.0131305>



Chaos
Special Topic: Nonlinear Model
Reduction From Equations and Data
Submit Today!

Reconstructing in-depth activity for chaotic 3D spatiotemporal excitable media models based on surface data



Cite as: Chaos 33, 013134 (2023); doi: 10.1063/5.0126824

Submitted: 17 September 2022 · Accepted: 22 December 2022 ·

Published Online: 23 January 2023



View Online



Export Citation



CrossMark

R. Stenger,^{1,2} S. Herzog,^{1,3,4,a)} I. Kottlarz,^{1,2} B. Rüdhardt,^{1,4} S. Luther,^{1,4,5} F. Wörgötter,³ and U. Parlitz^{1,2,4}

AFFILIATIONS

¹Max Planck Institute for Dynamics and Self-Organization, Am Fassberg 17, 37077 Göttingen, Germany

²Institute for the Dynamics of Complex Systems, University of Göttingen, Friedrich-Hund-Platz 1, 37077 Göttingen, Germany

³Department for Computational Neuroscience, Third Institute of Physics—Biophysics, University of Göttingen, 37077 Göttingen, Germany

⁴German Center for Cardiovascular Research (DZHK), partner site Göttingen, Robert-Koch-Str. 42a, 37075 Göttingen, Germany

⁵Institute of Pharmacology and Toxicology, University Medical Center Göttingen, Robert-Koch-Str. 40, 37075 Göttingen, Germany

^{a)}Author to whom correspondence should be addressed: sherzog3@gwdg.de

ABSTRACT

Motivated by potential applications in cardiac research, we consider the task of reconstructing the dynamics within a spatiotemporal chaotic 3D excitable medium from partial observations at the surface. Three artificial neural network methods (a spatiotemporal convolutional long-short-term-memory, an autoencoder, and a diffusion model based on the U-Net architecture) are trained to predict the dynamics in deeper layers of a cube from observational data at the surface using data generated by the Barkley model on a 3D domain. The results show that despite the high-dimensional chaotic dynamics of this system, such cross-prediction is possible, but non-trivial and as expected, its quality decreases with increasing prediction depth.

© 2023 Author(s). All article content, except where otherwise noted, is licensed under a Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>). <https://doi.org/10.1063/5.0126824>

Despite recent advances in the development of more powerful measurement devices, there are still quantities of interest that are difficult or impossible to access experimentally. An example of this limitation is the electrical activity in the heart muscle, which drives its pumping function and may be governed by (chaotic) spiral waves in the case of cardiac arrhythmia. A possible approach to fill this gap could be a reconstruction of the electrophysiological dynamics inside the heart (muscle) from available data, e.g., at the surface of the heart or the body. To address the general question of whether such a reconstruction is possible even in the case of strongly chaotic dynamics, we study a generic case given by a 3D chaotic Barkley model and compare the performance of three advanced machine learning algorithms used for this task. The prediction results obtained suggest that reconstruction of hidden cardiac activity may be feasible in future diagnostics.

I. INTRODUCTION

Observations of spatially extended dynamical systems may be limited in the sense that there are regions of space where direct measurements are difficult and expensive, or not possible at all. Such limitations apply, for example, in geophysics, where more effort is required to obtain data from deeper layers or remote areas that are not easily accessible. In medical applications, electrophysiology in the heart muscle is another example.¹ While mechanical contractions can be studied in 3D with high-speed ultrasound,² the corresponding electrical excitation waves cannot be measured directly, but only by analyzing (multichannel) electrocardiograms (ECGs) measured on the body surface or, in the case of removed animal hearts in Langendorff perfusion, by visualizing the electrical activity on the heart surface with the aid of fluorescent dyes in a process called optical mapping.^{3,4} Direct experimental observation

of the electrical excitation waves in the myocardium that control the mechanical contraction of the heart is not yet possible. This limitation is particularly a problem in the study of cardiac arrhythmias such as life-threatening (ventricular) fibrillation, in which spatiotemporal chaotic dynamics prevents effective pumping. Therefore, the general question arises whether it is in principle possible to estimate or reconstruct the dynamics inside the myocardium from surface data obtained *ex vivo* by optical mapping, or *in vivo* by multichannel ECG time series, or from (ultrasound) measurements of mechanical contraction of the heart.^{1,5–7} In what follows, we address the former task in a more abstract dynamical systems context using simulation data generated using a 3D Barkley model with chaotic dynamics.

Recent studies of the task of reconstructing the excitation dynamics in deeper cardiac layers employed and adapted data assimilation techniques from meteorology where the evolution of a physics-based model is corrected by incoming observations (here: from the surface).^{8–11} This approach was successfully applied to synthetic data generated by the Fenton–Karma model and the algorithms' sensitivity to model errors and other perturbations has been investigated.

Data assimilation methods require a physical model that is used to evolve the state of the system, which is then corrected using new observational data. An alternative are purely data-driven models trained on data sets representing the dynamics of interest to be studied. Such model-free approaches have proven to be very powerful in the recent past.^{12–14} For these approaches, data sets may be generated by some physical model or, if possible, obtained from experimental measurements. In general, such a data-driven approach can be motivated by delay embedding theorems^{15–17} suggesting that the measurement of some observables (for our current study: at the surface of a 3D cube) should in principle allow the reconstruction of all other variables of the system (here: any location within the cube). In practice, however, it is not clear whether this reasoning applies to this kind of extended (i.e., infinite dimensional) dynamical system and, even if it does, how large the embedding dimensions must be, if the system is dominated by a rather high-dimensional attractor (which is the case for the 3D Barkley model used here). Therefore, to investigate the practical limitations of cross-prediction, we applied three different machine learning architectures known for their ability to learn highly complex input–output relationships, using training and test data generated by a chaotic 3D Barkley model. The first variable u of the Barkley model corresponds in its dynamical role to the transmembrane voltage of cardiac cells. Therefore, this setup can be viewed as an idealized and simplified test case for the general prediction task “from the surface into depth” in extended excitable media, which serves to investigate the feasibility and limitations of machine learning methods for this reconstruction task. Motivated by the ongoing successes of machine learning in many non-linear dynamics fields: like fluid mechanics,¹⁸ protein structure prediction¹⁹ or model reduction and system identification,²⁰ we investigated three different machine learning approaches, presented in Sec. II B, as a solution to the prediction task “from the surface into depth.” We will show that the desired reconstruction is possible in principle for layers not too far from the surface, but requires sophisticated and high-performance algorithms to achieve this nontrivial goal.

II. METHODS

In the following Sec. II A, we will describe the data set which serves as a test case in this study. Then, in Sec. II B, we will introduce the machine learning algorithms used to predict the excitation dynamics in deeper layers of the data set from surface data.

A. Generation of 3D chaotic excitation waves using the Barkley model

The data set used in the following for training and testing different data-driven prediction algorithms was generated by the (cubic) Barkley model^{21–23}

$$\begin{aligned}\frac{\partial u}{\partial t} &= D \cdot \nabla^2 u + \frac{1}{\varepsilon} u(1-u) \left(u - \frac{v+b}{a} \right), \\ \frac{\partial v}{\partial t} &= u^3 - v,\end{aligned}\quad (1)$$

in 3D with no-flux boundary conditions.¹⁷ The fast variable u of this generic model of an excitable medium is diffusively coupled in space and corresponds to the transmembrane voltage in a cardiac- or neural context. Therefore, this variable is in the following assumed to be observable and sampled in space and time. For parameter values $a = 0.75$, $b = 0.06$, $\varepsilon = 0.08$, and $D = 0.02$, Eq. (1) generates chaotic spiral waves as illustrated in Fig. 1.

The 3D-simulations of the Barkley model Eq. (1) were conducted on a cube, discretized by a regular grid. To discretize the Laplace operator ∇^2 in Eq. (1), a 7-point stencil

$$\begin{aligned}(\nabla_7^2 u)_{i,j,k} &= [u_{i-1,j,k} + u_{i+1,j,k} + u_{i,j-1,k} + u_{i,j+1,k} \\ &\quad + u_{i,j,k-1} + u_{i,j,k+1} - 6u_{i,j,k}] / \Delta s^2\end{aligned}$$

was used where the indices i , j , and k stand for the discrete positions in the cube and $\Delta s = 0.1$ for the spatial discretization. While detailed studies of particular features of excitable (cardiac) media would benefit from using higher order approximations of the Laplace operator using 19-point or 27-point stencils,^{24,25} the current task to generate chaotic scroll-wave data does not require high fidelity simulations. Therefore, we chose here the 7-point stencil to keep the generation of data as simple as possible. The resulting system of ODEs was solved by the explicit Euler method with time steps of length $\Delta t = 0.01$ and no-flux boundary conditions.

To generate initial conditions that result in a chaotic evolution, the cube was divided into equilateral sub-cubes of size $6 \cdot \Delta s$ in each dimension, where all grid points in each of the generated sub-cubes of size $6 \times 6 \times 6$ grid points had the same u - and v -values, respectively. The values for u were randomly chosen between 0 and 1 from a uniform distribution. If the value exceeded 0.4 in a sub-cube, v was set to 1.0 in the same sub-cube. If this was not the case, v was set to 0.

Using this procedure for initialization, 2859 individual simulations were performed to generate training data using randomly chosen transient times between 3000 and 3500 Euler time steps of size $\Delta t = 0.01$. After that initial phase, the u value was recorded every 16th time step until 32 of such recordings had taken place. The effective sampling time was thus $t_S = 16 \cdot \Delta t = 0.16$ and the sampled trajectory segments cover a period of time of $32t_S = 5.12$ which corresponds to 512 Euler steps. Although it is possible that

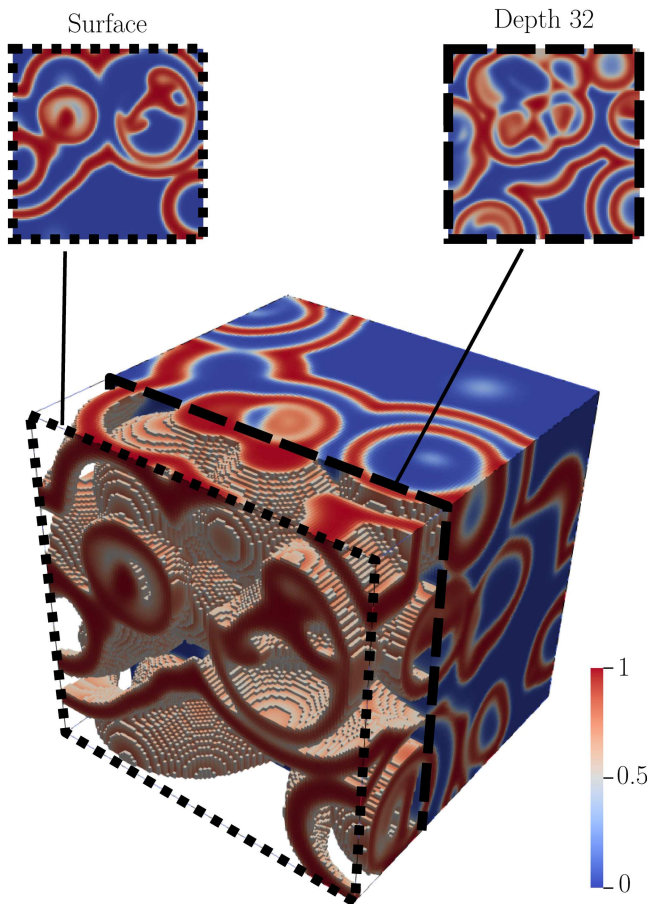


FIG. 1. Snapshot of the simulation of the Barkley model equation (1) showing the u variable on a regular grid of size $120 \times 120 \times 120$. For better visibility, the voxels with $u < 0.5$ are not displayed up to a depth of 32 voxels.

the dynamics is not chaotic after the initial phase, we did not observe this, so no simulations were subsequently discarded.

For a more intuitive understanding of the time span, we introduce a characteristic time T_c . We determine T_c by first calculating the Fourier transform of the time series of each voxel and average over all Fourier amplitude spectra. Then, the dominant frequency of the resulting averaged spectrum is determined which is the frequency with the largest amplitude. The period length corresponding to the dominant frequency is used as characteristic time scale T_c . For such calculation, we simulate 16 Barkley systems within the cube, which last over an extended time of 8192 time steps to reduce statistical fluctuations. This ensures that the simulation time exceeds the characteristic time by a multiple. The value of T_c can be regarded as an average period length of the signal in each voxel and corresponds in the chaotic Barkley system to 585 Euler steps of size Δt . Therefore, the maximal regarded time, covered by the input time series which last 512 Euler steps, is slightly shorter than the average period length.

At each sampling point in time from the 3D regular grid with $120 \times 120 \times 120$ nodes, only the first 32 layers in depth were stored

for further use. As a result for each initial condition, we obtained an array \mathbf{X}^* with $32 \times 120 \times 120 \times 32$ elements, where the first dimension describes the (sampling) time and the last three ones the spatial resolution (height, width, depth). Due to the reflection symmetries of the cube, any of its 6 surface can be considered as input and therefore, any chaotic trajectory generates six of the sequences \mathbf{X}^* that were all included in the data set.

A slice at time t and depth d is denoted by $\mathbf{x}_{t,d}$. The input for the artificial neural networks (ANNs) studied in this work are time series at the surface $\mathbf{x} = \{\mathbf{x}_{t,0}\}$ for $t = 0, \dots, 31$ where \mathbf{x} is an array with $32 \times 120 \times 120$ elements, representing a delay embedding forward in time. The desired output in a deeper layer $\mathbf{y} = \mathbf{x}_{0,d}$ for a given $d > 0$ is an array with 120×120 elements. The task of the data-driven machine learning is to find the mapping $\mathcal{N} : (32 \times 120 \times 120) \rightarrow (120 \times 120)$, such that $\mathcal{N}(\mathbf{x}) \approx \mathbf{y}$.

B. Neural network architectures

Within the framework of this study, three approaches have been investigated for the reconstruction task: A spatiotemporal (ST) spatiotemporal convolutional long-short-term-memory²⁶ (called ST-LSTM), an autoencoder²⁷ (AE), and a diffusion model^{28–30} (DM) based on a U-Net³¹ architecture. The ST-LSTM reconstructs individual layers using its internal states and an input representation, coming from a recurrent encoder. The AE approach predicts a block of several layers at once and the DM reconstructs exactly one slice. Therefore, for the ST-LSTM and DM approach, 32 individual networks are trained, each for one specific layer. By contrast, only one network needs to be trained for the AE. In the following, the three ANNs are introduced.

1. ST-LSTM

The ST-LSTM is based on the LSTM neural network architecture.³² An LSTM is a recurrent neural network, which is designed to iteratively process temporal sequences. Fundamentally, it relies on the (so-called) LSTM cell, which is a mathematical function that outputs “states,” based on individual time steps of the input time series. States usually divide into the actual cell state C , and the hidden state h . Those are iteratively updated with each time step as inputs. In the LSTM, connection weights are updated during the learning process such that the network’s behavior represents the underlying time series. In Fig. 2 (left), the computational graph of an LSTM is visualized, which is developed with each step of the time series \mathbf{x} . The figure shows an example of a stacked LSTM with three layers, where, first, the states C and h are developed over time and then passed on to serve as input of another LSTM cell at the respective time step. In contrast to the computational graph of the stacked LSTM, the ST-LSTM has an additional state M , as visualized in Fig. 2 (right). The M -state is, at first, developed through the layers of a given time step and then passed from its top layer to the bottom layer of the next time step. Therefore, in a stacked ST-LSTM, the M -state is not exclusively evolved in time and then transmitted to the next layer, such as the h -state. The ST-LSTM is based on the work of Wang *et al.*³² who motivate this approach by the fact that memory cells that belong to a stack of layers are in the original LSTM mutually independent and updated merely in the time domain. Under these circumstances, the bottom layer would totally ignore what

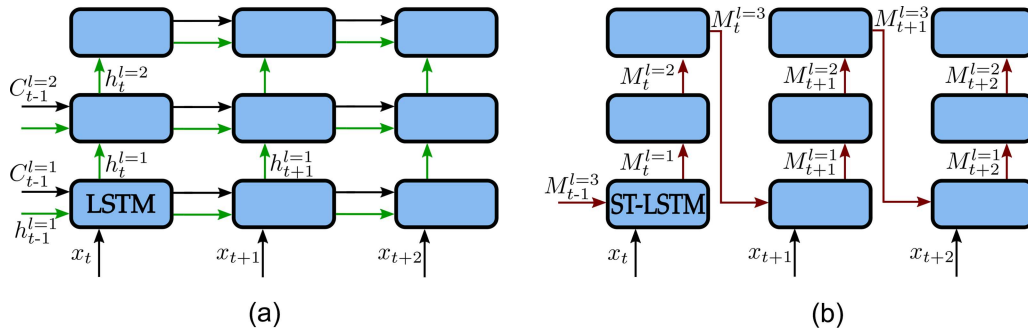


FIG. 2. Comparison of the computational flow within a stacked LSTM and a stacked ST-LSTM. The red arrows (right) mark the additional computations of the M -state and its computational flow. (a) Long-short-term-memory (LSTM). (b) Spatiotemporal LSTM (ST-LSTM).

had been memorized by the top layer in the previous time step. This caveat can be mended by propagating M -states. The ST-LSTM model in this work consists of a decoder and encoder part, where each of the parts consists of a stacked convolutional neural network with an ST-LSTM (see Fig. 3). After passing the whole sequence \mathbf{x} to the encoder, a representation in a form of the states h , C and M with fixed dimensionalities is obtained. The decoder then uses h , C and M as state inputs and the h state as time step input to reconstruct \mathbf{y} iteratively. The convolutional networks are supposed to down and upscale the inputs and outputs of the ST-LSTMs.

2. Convolutional autoencoder

The convolutional autoencoder (CAE) is a feed forward network that consists of a fully convolutional encoder and decoder, connected by dense layers. Given a surface time series $\{\mathbf{x}_t\}_{t \in \{0, \dots, 31\}}$, the autoencoder learns a low-dimensional representation in the code-block (Fig. 4, middle), from which the depths $\{\mathbf{y}_d\}_{d \in \{0, \dots, 31\}}$ at time $t = 0$ can be reconstructed. A scheme of the architecture is sketched in Fig. 4.

The encoder and decoder consist of three sub-blocks each, where each sub-block contains a (de-)convolutional layer, a batch normalization layer and a ReLU activation. In the encoder, three-dimensional convolutions take place in one temporal and two spatial

dimensions of the surface time series with a $4 \times 4 \times 4$ kernel and a $2 \times 2 \times 2$ stride. The decoder then re-generates the three spatial dimensions of the model output.

3. Diffusion model

For the diffusion model, we consider the reconstruction problem as an application of Bayes’ rule. Let \mathbf{y}_d be the non-observable quantity at some desired depth d and \mathbf{x} the observable quantity. The task is to reconstruct \mathbf{y}_d based on \mathbf{x} . Assuming that there is a connection between \mathbf{y}_d and \mathbf{x} , an observation distribution $p(\mathbf{x}|\mathbf{y}_d)$ is considered. If $p(\mathbf{x}|\mathbf{y}_d)$ and $p(\mathbf{y}_d)$ are known, then \mathbf{y}_d can be reconstructed by sampling from $p(\mathbf{y}_d|\mathbf{x})$. Since the prior distribution $p(\mathbf{y}_d)$ is unknown an artificial neural network shall be utilized as a generative model to approximate the prior distribution by training it on a dataset $\{\mathbf{y}_d^{(1)}, \dots, \mathbf{y}_d^{(N)}\} \sim p(\mathbf{y}_d)$, where $N = 15\,439$ is the number of training samples. Given the estimation of $p(\mathbf{y}_d)$ and the observation distribution $p(\mathbf{x}|\mathbf{y}_d)$, the posterior distribution $p(\mathbf{y}_d|\mathbf{x})$ can be determined through using Bayes, s.t.:

$$p(\mathbf{y}_d|\mathbf{x}) = \frac{p(\mathbf{x}|\mathbf{y}_d)p(\mathbf{y}_d)}{p(\mathbf{x})}$$

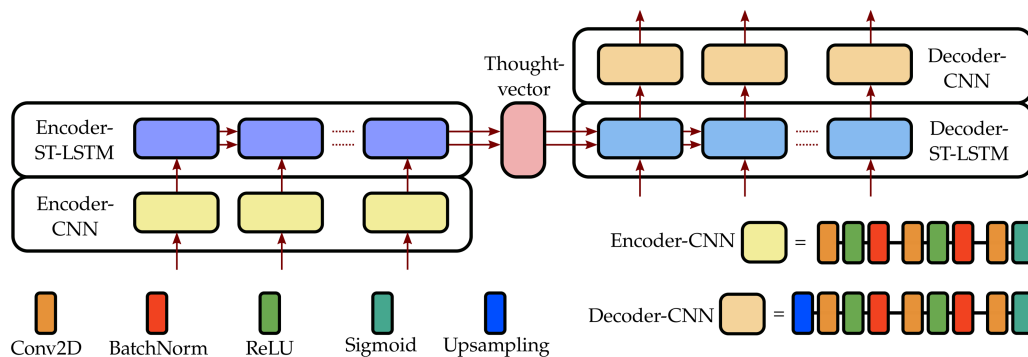


FIG. 3. Scheme of the LSTM model with an encoder–decoder structure consisting of CNNs and ST-LSTM’s.

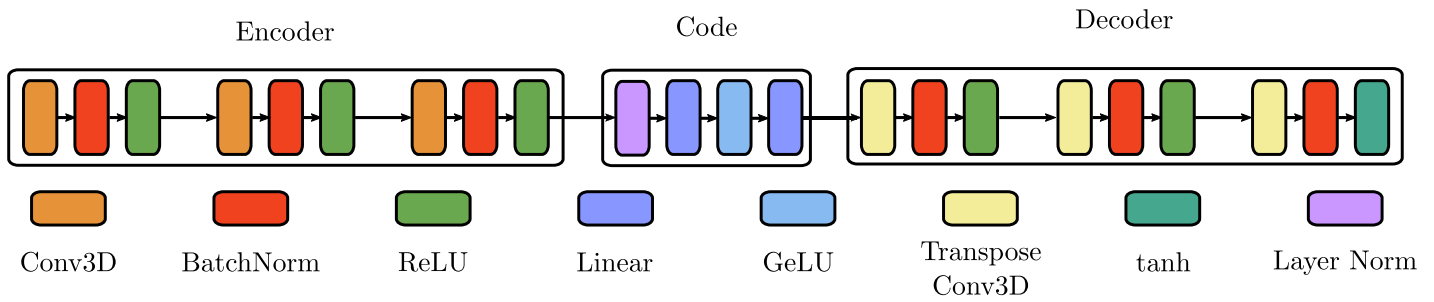


FIG. 4. Scheme of the autoencoder consisting of a fully convolutional encoder and decoder.

In the context of this work \mathbf{x} is not a measurement at one fixed time t but a sequence of measurements $\mathbf{x}_{t_1}, \mathbf{x}_{t_2}, \dots$. If these measurements were independent their joint distribution would be the product of marginal distributions. However, this assumption would only be valid if the Δt between two measurements were large enough, which is not the case here. Instead, a multilayer perceptron with one hidden layer is used to approximate $\{p(\mathbf{x}_1), \dots, p(\mathbf{x}_T)\} \mapsto p(\mathbf{x})$. To approximate the prior distribution $p(\mathbf{y}_d)$ a score based generative model^{33,34} based on the U-Net architecture³¹ is used. This allows an iterative sampling with a controllable generation conditioned on \mathbf{x} . To learn a sampling of \mathbf{y}_d from $p(\mathbf{y}_d)$, a stochastic process on a fixed time interval $[0, 1]$ is used. By applying the stochastic process to the training data \mathbf{y} , they become successively noisy. This progression is illustrated in Fig. 5. A fundamental part of the approach adopted here is to train the ANN to invert this process. In this way, a generator is obtained which, starting from a noisy input field generates a field which looks like a snapshot of the Barkley model. In order not to generate just any random snapshots of the dynamics, we continue to condition our network.

For this time information is incorporated via Gaussian random features, based on a so-called Gaussian Fourier Projection, where for a time step $t \in [0, 1]$ the corresponding Gaussian random feature is defined as $[\sin(2\pi\omega t); \cos(2\pi\omega t)]$, where $[\cdot; \cdot]$ denotes the concatenation of two vectors and ω a randomly sampled weight for a normal distribution with an expected value around a trained expected value and a trained variance. In this way, the ANN is trained not only to reconstruct some random snapshot of the Barkley model but to reconstruct the desired slice according to the input. For sampling, an exponential moving average of weights of the ANN was implemented. The architecture for the ANN is based on a U-Net³¹

architecture as shown in Fig. 6, which has been extended to include Gaussian random features. A further modification compared to the standard U-Net is the use of smoothed rectified linear unit activation functions³⁵ (SmeLU). The final sampling from the network is implemented analogously to Song *et al.*³³

C. Implementation details

1. Spatiotemporal long short-term memory

The ST-LSTM was implemented with PyTorch.³⁶ Considering each time step when computing the gradient for backpropagation through time leads to linear growth of the needed memory with the number of time steps. Therefore, a batch size of 4 was the maximum possible, while having a memory allocation below the available 32GB. For training, a learning rate scheduler was implemented, such that the learning rate was reduced by the factor of 3, as soon as the loss did not decrease during 512 iterations. The loss function was the mean squared error. The data were augmented by randomly rotating each sample either 0, 90, 180, or 270 degrees along the normal in each epoch.

2. Autoencoder

The autoencoder was implemented with PyTorch.³⁶ The network was trained using the general-purpose layer-wise adaptive large batch optimizer³⁷ (LAMB). Training was stopped when the validation loss did not improve for five consecutive epochs and the model state from the epoch with the lowest validation loss was saved. No data augmentation was used. The training loss function was the mean squared error.

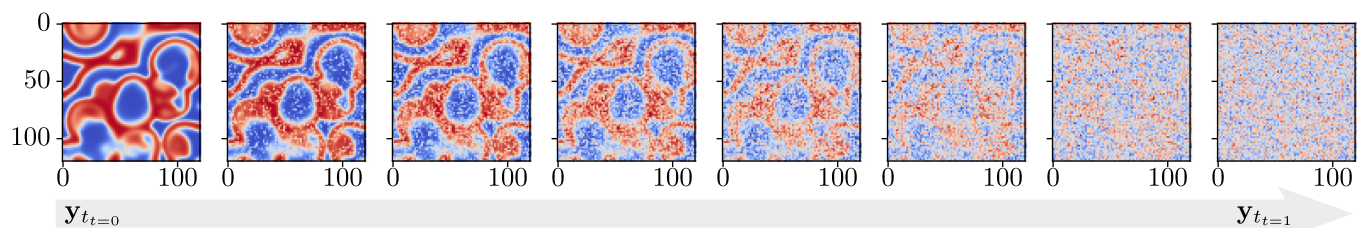


FIG. 5. Perturbation of \mathbf{y} , in the time interval $[0, 1]$ to the point of Gaussian noise.

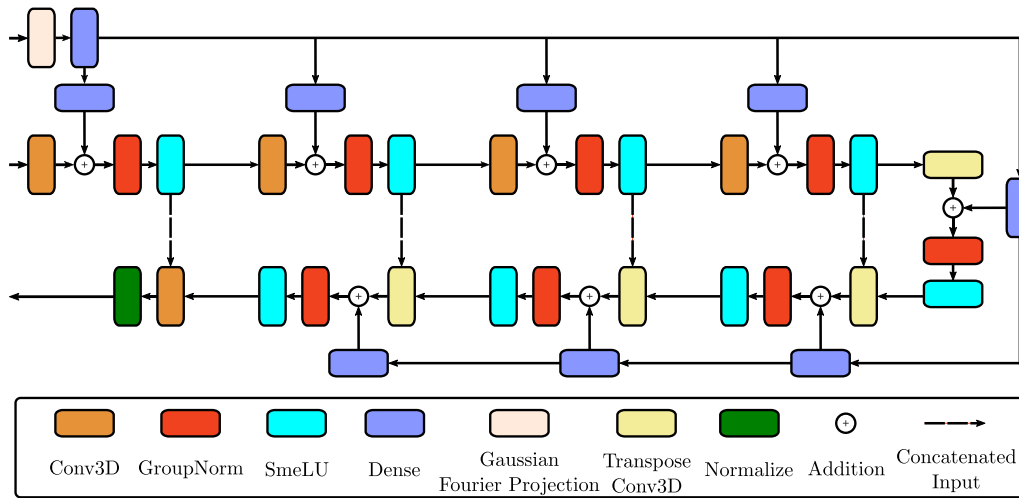


FIG. 6. U-Net based architecture for the score based model.

3. Diffusion model

The diffusion model was implemented with Jax,³⁸ using FLAX³⁹ for the ANN implementation, Optax³⁸ for the gradient method and Chex³⁸ as a utility. For each reconstruction depth, a separate network was trained. For training, early stopping with a patience of 25, CosineDecay⁴⁰ as learning rate schedule and the general-purpose layer-wise adaptive large batch optimizer³⁷ (LAMB) were used. No data augmentation was used. The sampling for inference was limited to 5 min.

III. RESULTS

To quantify the reconstruction performance of the different approaches, we calculate the mean absolute error between the predicted values $\mathcal{N}(\mathbf{x})$ and the true values \mathbf{y} , such that the absolute error for some sample i is given by: $e^{(i)} = |\mathcal{N}(\mathbf{x}^{(i)}) - \mathbf{y}^{(i)}|$ and the mean absolute error for an entire set of data with i samples ($i = 1, \dots, N$) by $MAE = \frac{1}{N} \sum_{i=1}^N e^{(i)}$. In the following section, we will show how the performance of the approaches depends on the input length.

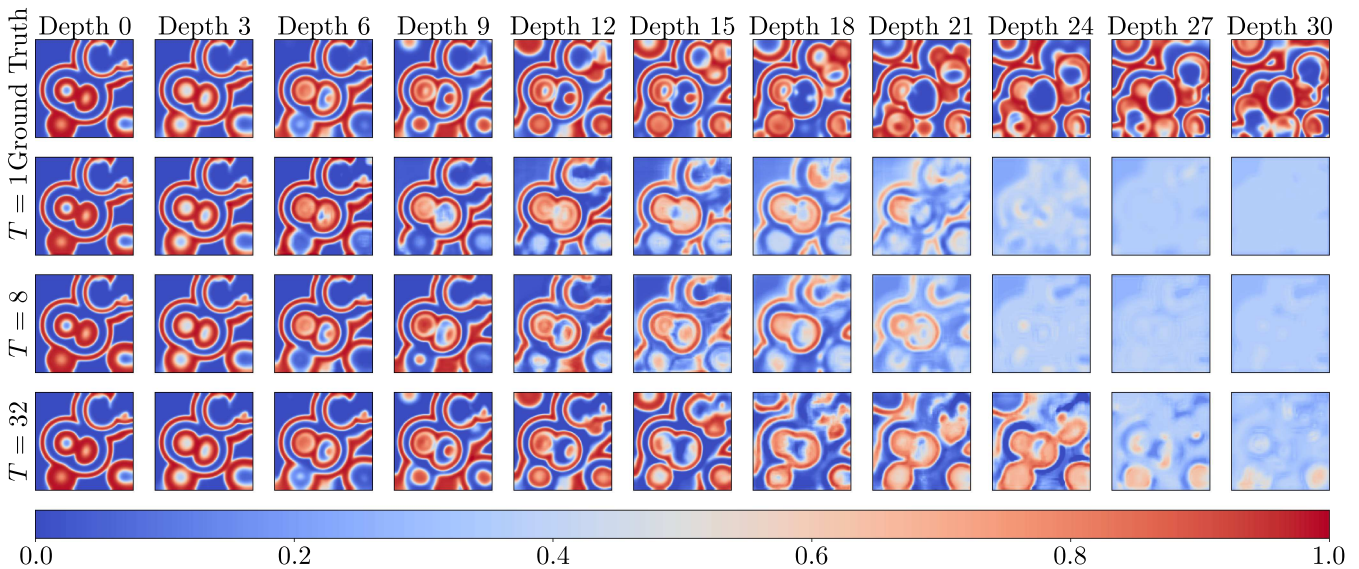


FIG. 7. Reconstruction with the ST-LSTM model approach on the data set with different input lengths $T \in \{1, 8, 32\}$ (rows 2-4). In the first row the ground truth panels for the different depths can be seen.

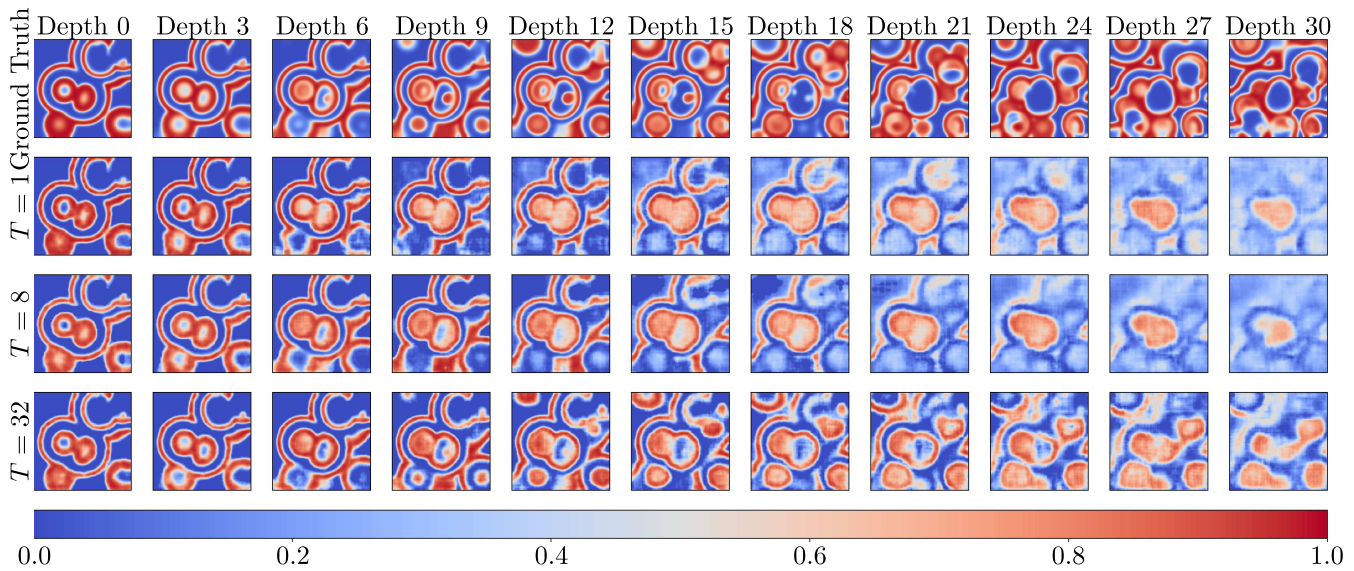


FIG. 8. Reconstruction from the autoencoder model approach on the data set with different input lengths $T \in \{1, 8, 32\}$.

A. Reconstructing in-depth activity with varying number of input time steps

To show this, each of the three approaches was trained with different input lengths, where a completely new network was trained for each input length. We find that, in general, the reconstruction performance increases by including more input time steps. However, performance gain decreases with more input time steps, while computational costs increases. However, there are a few

special characteristics that should be mentioned, too. Starting with the ST-LSTM:

Figure 7 visualizes the predictions of several ST-LSTMs, which are trained with different numbers of time steps ($T = 1$, $T = 8$ and $T = 32$). A trend, that the performance improves with increasing number of time steps, is clearly visible. However, finally all reconstructions approach some global mean value. This effect starts early for cases with fewer input time steps and the u values soon become increasingly blurry. The predictions of the convolutional

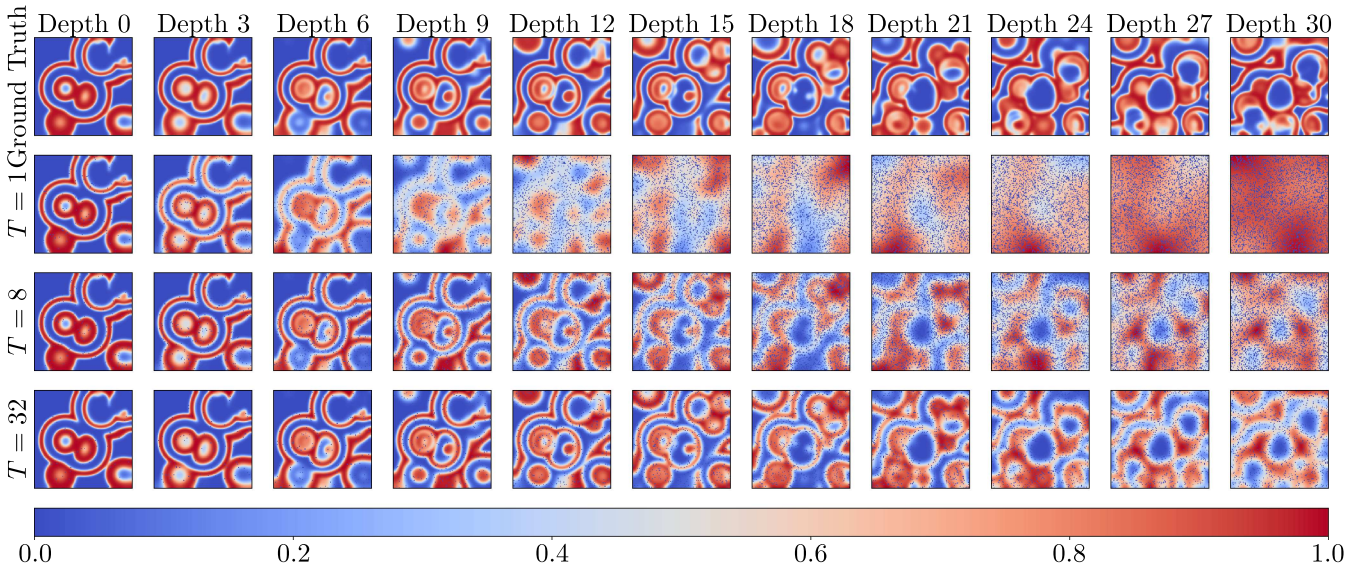


FIG. 9. Reconstruction from the diffusion model approach on the data set with different input lengths of $T \in \{1, 8, 32\}$.

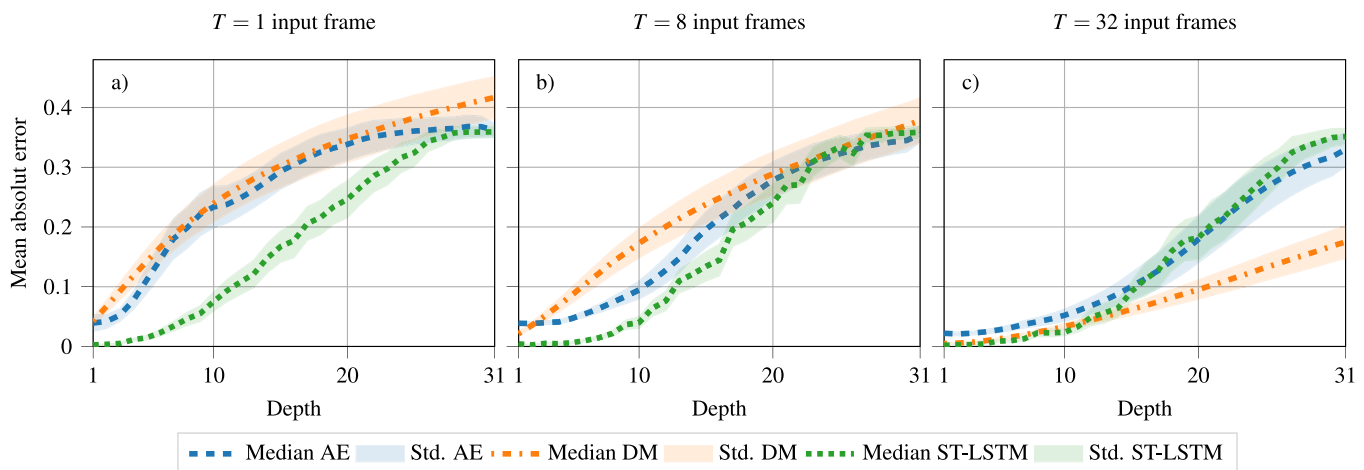


FIG. 10. Each diagram shows the statistical evaluation of reconstruction errors as a function of depth, depending on the input length (from left to right we have an input length of $T = 1, 8$, and 32). This evaluation was performed on the test data set which was not used during training. The lines represent the median value of the mean absolute errors (MAEs) and the bright areas represent the standard deviation.

autoencoder are displayed in Fig. 8. The behavior of the autoencoder is quite similar to that of the ST-LSTM and the autoencoder also tends to predict the mean value of the set with growing depth, especially for the shorter input time series ($T = \{1, 8\}$).

In the case of the diffusion model (DM), it can be seen in Figs. 9 and 10 that it benefits most from a long input and at the same time reconstructs worst when only a very short input is available. If an input length of 32 time steps is used, though, the DM approach is capable of recovering coarse structures even at a depth of 32. At the same time, one sees that the sampling approach based on³³ does not always lead to all pixels in the images being sampled, which suggests that the sampling strategy should be further improved. However, in the context of this work, we do not see this as a big problem, since one could e.g., replace the non-sampled points by values of their nearest neighbors. It should also be noted that while the DM approach generates the highest quality reconstructions when $T=32$ is used as input length, it also has the highest demands on computational resources, in the sense of training and inference time. While all ANNs benefit from a longer input sequence, the performance boost tends to be very different. Figure 10 illustrates the influence of the input length on the reconstruction error. For the case where only one input frame is used [Fig. 10(a)], the ST-LSTM approach outperforms the AE and DM approach significantly. With an input length of $T = 8$ [Fig. 10(b)] the approaches achieve similar results when deeper layers are supposed to be reconstructed. However, if an input length of $T = 32$ [Fig. 10(c)] is used, the DM approach outperforms the AE and ST-LSTM approaches. We see this as an indicator that the DM approach benefits most from long input sequences while the ST-LSTM approach is preferable for short input sequences.

B. Computational effort estimates of the three models

Training the models had different demands on the computational resources. Considering the cases where 32 time steps

were used as an input the training of the AE is the most economical, it was done on an NVIDIA GeForce GTX 1660 SUPER with 6GB VRAM in approximately 90–120 min to reconstruct all depths at once (It should be noted that the AE approach can predict all depths at once.). In second place is the ST-LSTM model which was trained on two NVIDIA P100 with 16 GB VRAM. For each case the model was trained for around 5 h in total. The DM has the highest demands on the componential resources it was trained on an Nvidia RTX 3090 and an AMD Threadripper Pro 5995wx where for each case it took on average 403 h and 17 min \pm 6 h and 7 min.

IV. DISCUSSION

In this study, we have shown how surface dynamics can be used to predict the in-depth activity of a 3D chaotic excitable medium. While previous attempts to address this dynamical reconstruction task employed data assimilation techniques, we demonstrate with our simulations that ANNs are a useful alternative for predicting hidden chaotic dynamics from partial observations. In detail, the ANNs we used and evaluated have different features and advantages: The DM provides the best results for all layers. ST-LSTM and AE are somewhat similar in performance, although in direct comparison the AE is preferable for deeper layers, while the ST-LSTM is in our application slightly better than the AE up to a depth of 15. Common to all ANNs is the fact that deeper layers are more difficult to reconstruct because the correlation between the input and the desired output decreases the deeper one wants to predict.

Time efficiency is another aspect in the evaluation of performance, in particular for potential real-time reconstruction of cardiac dynamics. Although the DM shows better performance compared to the other two approaches, it still has the significant disadvantage that it takes much longer for calculating the reconstruction of a layer. The DM needs about 5 minutes per layer, while the two other ANNs with comparable hardware equipment are in the range of less than one second. Comparing DM, ST-LSTM and AE, the AE has the

advantage of predicting all layers at once, which could be especially important for time-critical applications.

To reconstruct a deeper layer at time t we used here future data from the surface, sampled at times $t, t + t_s, t + 2t_s, \dots, t + (T - 1)t_s$. Due to this use of future samples there is a delay (or latency) for the prediction of the target layer, that is increasing with T . Predictions with larger T are more precise as shown in Fig. 10, so in practice there is a trade-off between delay and accuracy of the prediction. Interestingly, for small T the ST-LSTM provides more exact reconstructions than the AE or the DM as shown in Fig. 10(a), while for a larger number of input frames T the DM is superior, in particular for deeper layers [see Fig. 10(c)]. Of course, for any real-time application the delay due to the use of future samples from the surface is a practical disadvantage and an input stream going backward in time would be desirable. While from a theoretical point of view (Takens theorem) this would in principle provide equivalent information about the state of the system it turned out that in practice prediction results are much worse when using past values as input.

It should also be noted that by using a regular 3D grid, the geometry on which the reconstruction was performed is very simple. The generalization of our approaches to a non-regular grid, typical of a heart geometry, leads to completely new challenges that require different degrees of adaptation of the architecture for the ANNs selected here. The ST-LSTM and AE are based (in part) on convolutions, which can only be performed in this form on a regular grid. There are approaches, which deal with the adaptation of convolutions to non-regular grids, however, at this time no statements can be made on the performance of ANNs operating on irregular heart geometry. Here the DM offers the advantage that it works with a vector embedding of the lattice coordinates and therefore, a generalization to an irregular grid is easier than with the AE or ST-LSTM.

Besides these technical issues, there are still many open questions to be solved before a reconstruction of electrical excitation dynamics in the heart can be considered robust and reliable enough for future clinical applications. The dynamical system used for generating the data shown here differs in many aspects from a real heart, because we considered not only a simple 3D grid, but also the idealized case of a homogeneous and isotropic medium which contains neither blood vessels, fatty tissue, scars, or other heterogeneities nor conduction fiber directions. These strong simplifications were made intentionally to first find out how good a reconstruction can be under ideal constraints at best. On the other hand, from a dynamical point of view the Barkley model seems to be “more chaotic” than many (realistic) heart models, like the TenTusscher model,⁴¹ describing chaotic fibrillation. Furthermore, the fact that we considered an equilateral cube instead of a more flat slab of tissue made the reconstruction task probably more difficult compared to a real heart muscle of 1–2 cm thickness. These aspects need to be further investigated in future studies. One way to overcome uncertainties due to unknown heterogeneity and other perturbations of the medium could be to use a large portfolio of training data obtained with a variety of geometries, substrates, and different heart models. Such an approach with an ensemble of data⁴² may also serve to provide more reliable estimates of the robustness and correctness of reconstruction results for deeper layers.

ACKNOWLEDGMENTS

We acknowledge financial support by the Göttingen Graduate Center for Neurosciences, Biophysics and Molecular Biosciences (GGNB), the German Centre for Cardiovascular Research (DZHK) e.V., and the Max Planck Society.

AUTHOR DECLARATIONS

Conflict of Interest

The authors have no conflicts to disclose.

Author Contributions

R.S. and S.H. have contributed equally to this work.

R. Stenger: Conceptualization (equal); Data curation (equal); Investigation (equal); Methodology (equal); Software (equal); Validation (equal); Visualization (equal); Writing – original draft (equal); Writing – review & editing (equal). **S. Herzog:** Conceptualization (equal); Data curation (equal); Investigation (equal); Methodology (equal); Software (equal); Validation (equal); Visualization (equal); Writing – original draft (equal); Writing – review & editing (equal). **I. Kottlarz:** Methodology (equal); Software (equal); Validation (equal); Visualization (equal); Writing – original draft (equal); Writing – review & editing (equal). **B. Rüdhardt:** Writing – original draft (equal); Writing – review & editing (equal). **S. Luther:** Funding acquisition (equal); Supervision (equal); Writing – original draft (equal); Writing – review & editing (equal). **F. Wörgötter:** Funding acquisition (equal); Resources (equal); Supervision (equal); Writing – original draft (equal); Writing – review & editing (equal). **U. Parlitz:** Conceptualization (equal); Formal analysis (equal); Funding acquisition (equal); Project administration (equal); Resources (equal); Supervision (equal); Writing – original draft (equal); Writing – review & editing (equal).

DATA AVAILABILITY

The data for training and testing the approaches are publicly available at under <https://owncloud.gwdg.de/index.php/s/vD237B-DyaDG2OTa>.

REFERENCES

- ¹S. Alonso, M. Bär, and B. Echebarria, *Rep. Prog. Phys.* **79**, 096601 (2016).
- ²J. Christoph, M. Chebbok, R. Richter, J. Schröder-Schetelig, P. Bittihn, S. Stein, I. Uzelac, F. H. Fenton, G. Hasenfuß, R. F. Gilmour, and S. Luther, *Nature* **555**, 667–672 (2018).
- ³J. Christoph, J. Schröder-Schetelig, and S. Luther, *Prog. Biophys. Mol. Biol.* **130**, 150 (2017).
- ⁴J. Christoph, J. Schröder-Schetelig, U. Parlitz, and S. Luther, “Zipes and Jalife’s cardiac electrophysiology: From cell to bedside,” in *Electromechanic Imaging of the Heart* (Elsevier, 2021).
- ⁵J. Lebert and J. Christoph, *Chaos* **29**, 093117 (2019).
- ⁶J. Christoph and J. Lebert, *Chaos* **30**, 123134 (2020).
- ⁷S. Herzog, R. S. Zimmermann, J. Abele, S. Luther, and U. Parlitz, *Front. Appl. Math. Stat.* **6**, 616584 (2021).
- ⁸M. J. Hoffman, N. S. LaVigne, S. T. Scorse, F. H. Fenton, and E. M. Cherry, *Chaos* **26**, 013107 (2016).
- ⁹N. S. LaVigne, N. Holt, M. J. Hoffman, and E. M. Cherry, *Chaos* **27**, 093911 (2017).
- ¹⁰M. J. Hoffman and E. M. Cherry, *Philos. Trans. R. Soc. A* **378**, 20190388 (2020).

- ¹¹C. D. Marcotte, F. H. Fenton, M. J. Hoffman, and E. M. Cherry, *Chaos* **31**, 013118 (2021).
- ¹²J. Jiang and Y.-C. Lai, *Phys. Rev. Res.* **1**, 033056 (2019).
- ¹³S. Herzog, F. Wörgötter, and U. Parlitz, *Chaos* **29**, 123116 (2019).
- ¹⁴J. Lebert, M. Mittal, and J. Christoph, “Reconstruction of three-dimensional scroll waves in excitable media from two-dimensional observations using deep neural networks,” [arXiv:2209.06860](https://arxiv.org/abs/2209.06860) (2022).
- ¹⁵F. Takens, in *Dynamical Systems and Turbulence, Warwick 1980*, edited by D. Rand and L.-S. Young (Springer, 1981), pp. 366–381.
- ¹⁶T. Sauer, J. A. Yorke, and M. Casdagli, *J. Stat. Phys.* **65**, 579 (1991).
- ¹⁷A. Datsersis and U. Parlitz, *Nonlinear Dynamics—A Concise Introduction Interlaced with Code* (Springer, 2022).
- ¹⁸S. L. Brunton, B. R. Noack, and P. Koumoutsakos, *Annu. Rev. Fluid Mech.* **52**, 477 (2020).
- ¹⁹J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Židek, A. Potapenko, A. Bridgland, C. Meyer, S. A. A. Kohl, A. J. Ballard, A. Cowie, B. Romera-Paredes, S. Nikolov, R. Jain, J. Adler, T. Back, S. Petersen, D. Reiman, E. Clancy, M. Zielinski, M. Steinegger, M. Pacholska, T. Berghammer, S. Bodenstein, D. Silver, O. Vinyals, A. W. Senior, K. Kavukcuoglu, P. Kohli, and D. Hassabis, *Nature* **596**, 583 (2021).
- ²⁰S. Klus, F. Nüske, S. Peitz, J.-H. Niemann, C. Clementi, and C. Schütte, *Phys. D: Nonlinear Phenom.* **406**, 132416 (2020).
- ²¹D. Barkley, M. Kness, and L. S. Tuckerman, *Phys. Rev. A* **42**, 2489 (1990).
- ²²D. Barkley, *Scholarpedia* **3**, 1877 (2008).
- ²³S. Berg, S. Luther, and U. Parlitz, *Chaos* **21**, 033104 (2011).
- ²⁴M. Dowle, R. Martin Mantel, and D. Barkley, *Int. J. Bifurcat. Chaos* **07**, 2529 (1997).
- ²⁵P. Bittihn, *Complex Structure and Dynamics of the Heart*, 1st ed. (Springer International Publishing, Switzerland, 2015).
- ²⁶Y. Wang, M. Long, J. Wang, Z. Gao, and P. S. Yu, in *Advances in Neural Information Processing Systems*, edited by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Curran Associates, Inc., 2017), Vol. 30.
- ²⁷G. E. Hinton and R. R. Salakhutdinov, *Science* **313**, 504 (2006).
- ²⁸J. Sohl-Dickstein, E. A. Weiss, N. Maheswaranathan, and S. Ganguli, *CoRR abs/1503.03585* (2015).
- ²⁹Y. Song and S. Ermon, *CoRR abs/1907.05600* (2019).
- ³⁰J. Ho, A. Jain, and P. Abbeel, *CoRR abs/2006.11239* (2020).
- ³¹O. Ronneberger, P. Fischer, and T. Brox, in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, edited by N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi (Springer International Publishing, Cham, 2015), pp. 234–241.
- ³²S. Hochreiter and J. Schmidhuber, *Neural Comput.* **9**, 1735 (1997).
- ³³Y. Song and S. Ermon, “Maximum likelihood training of score-based diffusion models,” in *Advances in Neural Information Processing Systems*, edited by M. Ranzato, A. Beygelzimer, Y. Dauphin, P. S. Liang, and J. Wortman Vaughan (Curran Assoc. Inc., 2019), Vol. 34, pp. 1415–1428.
- ³⁴Y. Song and S. Ermon, in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6–12, 2020, virtual*, edited by H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin (Curran Assoc. Inc., 2020).
- ³⁵G. I. Shamir and D. Lin, “Real world large scale recommendation systems reproducibility and smooth activations,” [arXiv:2202.06499](https://arxiv.org/abs/2202.06499) (2022).
- ³⁶A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, in *Advances in Neural Information Processing Systems 32*, edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett (Curran Associates, Inc., 2019), pp. 8024–8035.
- ³⁷Y. You, J. Li, S. Reddi, J. Hseu, S. Kumar, S. Bhojanapalli, X. Song, J. Demmel, K. Keutzer, and C.-J. Hsieh, “Large batch optimization for deep learning: Training bert in 76 minutes,” *International Conference on Learning Representations (2020)*; available at <https://openreview.net/forum?id=Syx4wnEtvH>.
- ³⁸I. Babuschkin, K. Baumli, A. Bell, S. Bhupatiraju, J. Bruce, P. Buchlovsky, D. Budden, T. Cai, A. Clark, I. Danihelka, C. Fantacci, J. Godwin, C. Jones, T. Hennigan, M. Hessel, S. Kapturowski, T. Keck, I. Kemaev, M. King, L. Martens, V. Mikulik, T. Norman, J. Quan, G. Papamakarios, R. Ring, F. Ruiz, A. Sanchez, R. Schneider, E. Sezener, S. Spencer, S. Srinivasan, W. Stokowiec, and F. Viola, (2018). “JAX: Composable transformations of Python+NumPy programs,” <http://github.com/google/jax>.
- ³⁹J. Heek, A. Levskaya, A. Oliver, M. Ritter, B. Rondepierre, A. Steiner, and M. van Zee (2020). “Flax: A neural network library and ecosystem for JAX,” <http://github.com/google/flax>.
- ⁴⁰I. Loshchilov and F. Hutter, “Sgdr: Stochastic gradient descent with warm restarts,” in *International Conference on Learning Representations (2017)*; available at <https://openreview.net/forum?id=Skq89Scxx>.
- ⁴¹K. H. W. J. Ten Tusscher, D. Noble, P. J. Noble, and A. V. Panfilov, *Am. J. Physiol.-Heart Circ. Physiol.* **286**, H1573 (2004).
- ⁴²H. Lilienkamp and T. Lilienkamp, *Sci. Rep.* **11**, 19767 (2021).