RTSNET: DEEP LEARNING AIDED KALMAN SMOOTHING

Xiaoyong Ni, Guy Revach, Nir Shlezinger, Ruud J. G. van Sloun, and Yonina C. Eldar

ABSTRACT

The smoothing task is the core of many signal processing applications. It deals with the recovery of a sequence of hidden state variables from a sequence of noisy observations in a one-shot manner. In this work we propose RTSNet, a highly efficient model-based and data-driven smoothing algorithm. RTSNet integrates dedicated trainable models into the flow of the classical Rauch-Tung-Striebel (RTS) smoother, and is able to outperform it when operating under model mismatch and non-linearities while retaining its efficiency and interpretability. Our numerical study demonstrates that although RTSNet is based on more compact neural networks, which leads to faster training and inference times, it outperforms the state-of-the-art, data-driven smoother in a non-linear use case.

***Index Terms*—** Kalman smoother, deep learning.

1. INTRODUCTION

Estimating a hidden state of a dynamical system from noisy observations is one of the most fundamental tasks in signal processing, with applications in localization, tracking, and navigation [1]. When all data is available at the beginning of the processing task we can get a more accurate state estimate using a *smoothing* algorithm, as opposed to *filtering*. While filtering (also known as real-time tracking) estimates the current state from past and current observations, smoothing is about simultaneous state recovery of all available data on the entire time horizon. Filtering and smoothing date back to the work of Wiener from 1949 [2].

The celebrated Kalman filter (KF) from the early 1960s [3] is a low complexity and theoretically sound algorithm for filtering in discrete-time. KF and its later non-linear variants [4, 5] are still considered to be the leading approaches for various real world applications. The Rauch-Tung-Striebel (RTS) smoother [6] from 1965 is considered the first algorithm for smoothing in discrete-time, and it is also the basis for multiple non-linear variants [1, Ch. 10]. The RTS smoother is also known as the Kalman smoother (KS), because it implements maximum likelihood estimation for linear Gaussian state space (SS) models by applying the KF followed by a recursive update step based on future observations. Despite its low complexity and theoretical soundness, applying the model-based (MB) KS in practical scenarios may be limited due to its critical dependence on accurate knowledge of the underlying SS model, which may be complex and difficult to characterize faithfully. The non-linear variants of the KS (e.g., extended KS) are not minimum mean-squared error (MMSE) optimal, and performance tends to degrade in the presence of strong non-linearities.

Data-driven (DD) approaches are an alternative to MB algorithms, relaxing the requirement for explicit and accurate knowledge of the SS model. Many of these strategies are now based on deep neural networks (DNNs), which have shown remarkable success in capturing the subtleties of complex processes and replacing the need to explicitly characterize the domain of interest [7, 8]. While DNNs such as recurrent neural networks (RNNs) [9, 10] and attention mechanisms [11] have been shown to perform very well for time series related tasks mostly in intractable environments, they do not incorporate domain knowledge such as structured SS models in a principled manner. These DD approaches thus require many trainable parameters and large data sets even for simple sequence models [12] and lack the interpretability of MB methods. From the large body of work that incorporates SS models with DNNs, e.g., [13–18], the one most directly related to smoothing is [19], which proposed an iterative algorithm on top of a hybrid graphical model that combines a MB module with a neural network (NN). This DD smoother learns from data to improve its performance when compared to the MB alone, but it involves multiple possibly lengthy iterations, resulting in high complexity and slow inference.

In this work we propose RTSNet, a hybrid MB/DD, which is an efficient recursive smoothing algorithm for (possibly) non-linear dynamics and partially known SS models. Our design is inspired by our previously proposed hybrid algorithms [20–23] and is built on top of KalmanNet [24, 25], a DNN-based KF, in the same way as the KS is built on top of the KF. By replacing the forward and backward Kalman gains (KGs) with dedicated compact RNNs and training it in a supervised manner, RTSNet retains the interpretability and the optimality of the MB KS with full domain knowledge, and notably outperforms it with model mismatch and strong non-linearities. RTSNet is shown to outperform the state-of-the-art DD smoother of [19], while using less trainable parameters

X. Ni and G. Revach are with the Institute for Signal and Information Processing (ISI), D-ITET, ETH Zürich, Switzerland (e-mail: xiaoni@student.ethz.ch, grevach@ethz.ch). N. Shlezinger is with the School of ECE, Ben-Gurion University of the Negev, Beer Sheva, Israel (e-mail: nirshl@bgu.ac.il). R. J. G. van Sloun is with the EE Dpt., Eindhoven University of Technology, and with Phillips Research, Eindhoven, The Netherlands (e-mail: r.j.g.v.sloun@tue.nl). Y. C. Eldar is with the Faculty of Math and CS, Weizmann Institute of Science, Rehovot, Israel (e-mail: yonina.eldar@weizmann.ac.il). The authors thank Prof. Hans-Andrea Loeliger for his helpful comments and discussion.

and relying on only partial knowledge of the SS model.

The rest of this paper is organized as follows: Section 2 formulates the DD smoothing problem; Section 3 details the proposed RTSNet; and Section 4 presents a numerical study.

2. SYSTEM MODEL

2.1. Data-Driven Smoothing Problem Formulation

We consider *fixed-interval* smoothing; i.e., the recovery of a state block $\{\mathbf{x}_t\}_{t=1}^T$ given a block of noisy observations $\{\mathbf{y}_t\}_{t=1}^T$ for a fixed length T. The state and the observations are related via a dynamical system represented by a non-linear, Gaussian, continuous SS model in discrete-time:

$$\mathbf{x}_t = \mathbf{f}(\mathbf{x}_{t-1}) + \mathbf{e}_t, \qquad \mathbf{e}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}), \quad \mathbf{x}_t \in \mathbb{R}^m, \tag{1a}$$

$$\mathbf{y}_t = \mathbf{h}(\mathbf{x}_t) + \mathbf{v}_t, \qquad \mathbf{v}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{R}), \quad \mathbf{y}_t \in \mathbb{R}^n. \tag{1b}$$

In (1), $\mathbf{f}(\cdot)$ and $\mathbf{h}(\cdot)$ are (possibly) non-linear functions, while $\mathbf{e}_t$ and $\mathbf{v}_t$ are Gaussian noise signals with covariance matrices $\mathbf{Q}$ and $\mathbf{R}$, respectively. Our objective is to design an algorithm that maps a block of observations into a block of state estimators $\{\hat{\mathbf{x}}_t\}_{t=1}^{T-1}$, given the initial $\mathbf{x}_0$ and final $\mathbf{x}_T$ states. We focus on scenarios where one has partial knowledge of the system model; namely, we know (or have an approximation of) $\mathbf{f}(\cdot)$ and $\mathbf{h}(\cdot)$ from a dynamical model, yet the noise statistics are not known. However, the system has access to a labeled data set comprised of a sequence of observations and their corresponding states. Our approach is thus based on utilizing the data via deep learning, combined with operation of the MB KS, reviewed next.

2.2. RTS Smoother

Here, we review the MB KS [6]. Since we consider non-linear SS models (1), we focus on the extended RTS smoother [1, Ch. 10], which utilizes the linear approximations of $\mathbf{f}(\cdot)$ and $\mathbf{h}(\cdot)$, denoted $\hat{\mathbf{F}}$ and $\hat{\mathbf{H}}$, respectively, obtained by computing the Jacobian matrices at the current estimated state.

The MB KS recovers the latent state variables using two linear recursive steps referred to as the forward and backward passes. The forward pass is a standard KF, which updates its *prior* (for state and covariance) based on past observations. For each t the forward pass computes $\hat{\mathbf{x}}_{t|t-1} = \mathbf{f}(\hat{\mathbf{x}}_{t-1})$ and $\hat{\mathbf{y}}_{t|t-1} = \mathbf{h}\left(\hat{\mathbf{x}}_{t|t-1}\right)$. while $\hat{\mathbf{\Sigma}}_{t|t-1} = \hat{\mathbf{F}} \cdot \hat{\mathbf{\Sigma}}_{t-1|t-1} \cdot \hat{\mathbf{F}}^\top + \mathbf{Q}$, and $\hat{\mathbf{S}}_t = \hat{\mathbf{H}} \cdot \hat{\mathbf{\Sigma}}_{t|t-1} \cdot \hat{\mathbf{H}}^\top + \mathbf{R}$. The predictions are updated via $\hat{\mathbf{x}}_{t|t} = \hat{\mathbf{x}}_{t|t-1} + \boldsymbol{\mathcal{K}}_t \cdot \Delta\mathbf{y}_t$ and $\hat{\mathbf{\Sigma}}_{t|t} = \hat{\mathbf{\Sigma}}_{t|t-1} - \boldsymbol{\mathcal{K}}_t \cdot \hat{\mathbf{S}}_t \cdot \boldsymbol{\mathcal{K}}_t^\top$, where $\Delta\mathbf{y}_t = \mathbf{y}_t - \hat{\mathbf{y}}_{t|t-1}$ and $\boldsymbol{\mathcal{K}}_t$ is the *forward* KG:

$$\boldsymbol{\mathcal{K}}_t = \hat{\mathbf{\Sigma}}_{t|t-1} \cdot \hat{\mathbf{H}}^\top \cdot \hat{\mathbf{S}}_t^{-1}. \tag{2}$$

The backward pass is carried out in a cascade to the forward pass, fusing an estimate based on future observations with the result of the KF. This is achieved by going over the time instances from $t = T-1$ to $t = 1$. For each t, the posterior is updated via

$$\hat{\mathbf{x}}_t = \hat{\mathbf{x}}_{t|t} + \boldsymbol{\mathcal{G}}_t \cdot \overleftarrow{\Delta}\mathbf{x}_{t+1}, \quad \overleftarrow{\Delta}\mathbf{x}_{t+1} = \hat{\mathbf{x}}_{t+1} - \mathbf{f}\left(\hat{\mathbf{x}}_{t|t}\right) \tag{3}$$

while the second-order moments are updated via $\hat{\mathbf{\Sigma}}_t = \hat{\mathbf{\Sigma}}_{t|t} - \boldsymbol{\mathcal{G}}_t \cdot \Delta\mathbf{\Sigma}_{t+1} \cdot \boldsymbol{\mathcal{G}}_t^\top$, with $\Delta\mathbf{\Sigma}_{t+1} = \hat{\mathbf{\Sigma}}_{t+1} - \hat{\mathbf{\Sigma}}_{t+1|t}$. Here, $\boldsymbol{\mathcal{G}}_t$ is the backward KG, computed from the forward pass as

$$\boldsymbol{\mathcal{G}}_t = \hat{\mathbf{\Sigma}}_{t|t} \cdot \hat{\mathbf{F}}^\top \cdot \hat{\mathbf{\Sigma}}_{t+1|t}^{-1}. \tag{4}$$

For a linear SS model, the estimate achieves the MMSE. However, it requires full knowledge of the underlying model and is notably degraded in the presence of model mismatch. When $\mathbf{f}(\cdot)$ and $\mathbf{h}(\cdot)$ are non-linear, their linear approximation is sub-optimal and limits the accuracy in highly non-linear setups, even when the SS model is known. These drawbacks motivate deriving a DNN-aided KS, as detailed below.

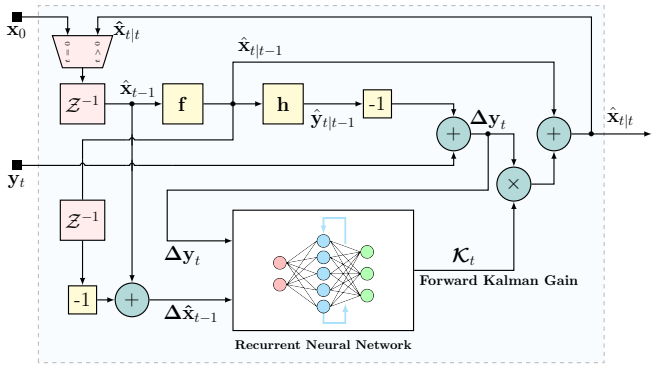
3. RTSNET

3.1. Architecture

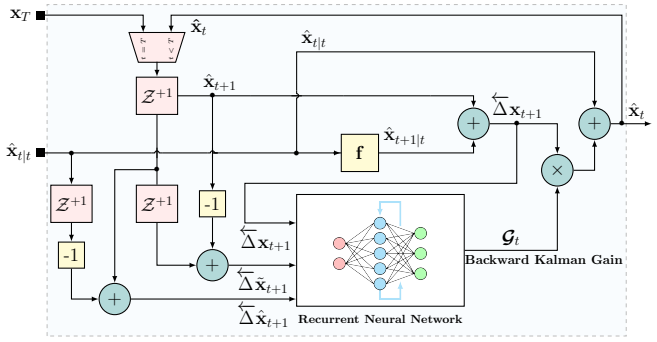
The basic design idea of RTSNet is to utilize the structure of the MB RTS smoother and to replace modules depending on unavailable domain knowledge (i.e., noise statistics and model mismatch) with trainable DNNs, which could be then trained in a supervised end-to-end manner from labeled data. The reason for choosing the RTS smoother as our backbone MB smoothing algorithm is because, as opposed to other alternatives, e.g., MBF [26] and BIFM [27], in RTS all the unknown domain knowledge that is required for computing the first-order statistical moment, i.e., the state estimate $\hat{\mathbf{x}}_t$, is encapsulated in the forward and backward KGs, $\boldsymbol{\mathcal{K}}_t$, and $\boldsymbol{\mathcal{G}}_t$, respectively. Since both KGs involve tracking time-evolving second-order moments, where (2) uses the statistics of both the state process and the measurements, while (4) utilizes only the state statistics, they are replaced by RNNs in RTSNet, with input features encapsulating the missing statistics.

The resulting RTSNet boils down to a highly efficient and interpretable recursive algorithm with forward and backward passes. The forward pass is built on KalmanNet [25] using architecture 2 of [25] for the RNN that computes $\boldsymbol{\mathcal{K}}_t$. This model includes three cascaded gated recurrent unit (GRU) layers with dedicated input and output fully connected (FC) layers. The input features are designed to capture differences in the state and the observation model, as these differences are mostly affected by unknown noise statistics.

The backward pass, illustrated in Fig. 1, implements (3). As noted above, $\boldsymbol{\mathcal{G}}_t$ depends on the statistics of $\mathbf{x}_t$, and particularly on its estimates provided by the forward pass. To compute $\boldsymbol{\mathcal{G}}_t$ in a learned manner, we utilize the following features, which are related to the unknown underlying statistics:
1. *Update difference* between the smoothing posterior and the forward prior: $\overleftarrow{\Delta}\mathbf{x}_{t+1} = \hat{\mathbf{x}}_{t+1} - \hat{\mathbf{x}}_{t+1|t}$.



(a) Forward pass.



(b) Backward pass.

Fig. 1: Block diagram of RTSNet.

2. *Backward forward difference* between the smoothing posterior and the forward prior: $\overleftarrow{\Delta}\hat{\mathbf{x}}_{t+1} = \hat{\mathbf{x}}_{t+1} - \hat{\mathbf{x}}_{t+1|t+1}$.
3. *Evolution difference* between two consecutive smoothing posteriors $\overleftarrow{\Delta}\tilde{\mathbf{x}}_{t+1} = \hat{\mathbf{x}}_{t+2} - \hat{\mathbf{x}}_{t+1}$.

The first two features capture the uncertainty in the state estimate, where the differences remove predictable components such that they are mostly affected by the unknown noise statistics. The third feature is related to the evolution of the predicted state, and thus reflects on its statistics that are tracked by the MB KS. The features are mapped into an estimate of $\boldsymbol{\mathcal{G}}_t$ using a two-layer GRU with input and output FC layers. Since the MB KS tracks the $m \times m$ matrix $\hat{\boldsymbol{\Sigma}}_t$ in computing $\boldsymbol{\mathcal{G}}_t$, we set the GRU hidden state size to m^2.

3.2. Training Algorithm

We train RTSNet end-to-end using labeled data to minimize the mean-squared error (MSE) between the predicted $\hat{\mathbf{x}}_t$ and the true state $\mathbf{x}_t$. The training set is comprised of N sequences of observations and their corresponding states, $\{(\mathbf{Y}_i, \mathbf{X}_i)\}_{i=1}^N$, where $\mathbf{Y}_i = \left[\mathbf{y}_1^{(i)}, \ldots, \mathbf{y}_{T_i}^{(i)}\right]$, $\mathbf{X}_i = \left[\mathbf{x}_0^{(i)}, \mathbf{x}_1^{(i)}, \ldots, \mathbf{x}_{T_i}^{(i)}\right]$, and T_i is the length of the ith sequence. Letting $\boldsymbol{\Theta}$ denote the overall trainable parameters of the learned KGs, the loss function is set to

$$\mathcal{L}(\boldsymbol{\Theta}) = \sum_{i=1}^{N} \frac{1}{T_i} \sum_{t=1}^{T_i} \left\| \hat{\mathbf{x}}_t\left(\mathbf{y}_t^{(i)}; \boldsymbol{\Theta}\right) - \mathbf{x}_t^{(i)} \right\|^2 + \gamma \cdot \|\boldsymbol{\Theta}\|^2, \quad (5)$$

where and γ is the weight decay coefficient. The gradient of the loss with respect to the RNN parameters is computed using backpropagation through time.

3.3. Discussion

The hybrid MB/DD operation of the proposed RTSNet allows it to enjoy the best of both worlds, with advantages over both the classical MB smoother as well as DD ones. RTSNet operates without explicit knowledge of the noise statistics, and learns to estimate its effects *indirectly* from the Kalman gain. In this way, it is able to avoid parameter estimation, which leads to an efficient training scheme and allows compensation for model mismatch; it avoids linearization and is less sensitive to non-linearities; and does not require inverting matrices while inferring rapidly with low computation complexity due to efficient RNNs. As shown in Section 4, RTSNet outperforms the MB smoother in the presence of model mismatches and harsh non-linearities, as well as the computationally intensive DD benchmark of [19]. This gain is achieved while RTSNet only makes two efficient passes on the data, while the benchmark if [19] makes one hundred message-passing iterations between nodes in the graph, and relies on knowledge of the SS model and an exhaustive grid search to optimize its MB part. The above makes RTSNet attractive for applications on hardware-limited devices, as exemplified for KalmanNet in [24], and allows it to achieve improved accuracy for limited data sets. Finally, while RTSNet utilizes a single learned forward-backward pass, it can be extended to carry out multiple passes via deep unfodling [20], which is expected to further improve performance, and is left for future investigation.

4. NUMERICAL EVALUATIONS

Here, we evaluate[1] RTSNet, both on a linear SS model and the Lorenz attractor, non-linear chaotic model. In the following, we set $\mathbf{Q} = \mathrm{q}^2 \cdot \mathbf{I}_m$ and $\mathbf{R} = \mathrm{r}^2 \cdot \mathbf{I}_n$, defining $\nu \triangleq \frac{\mathrm{q}^2}{\mathrm{r}^2}$.

4.1. Linear Model

We evaluate RTSNet on a linear SS model; i.e., there exist matrices $\mathbf{F}$ and $\mathbf{H}$ such $\mathbf{f}(\mathbf{x}) = \mathbf{F}\mathbf{x}$ and $\mathbf{h}(\mathbf{x}) = \mathbf{H}\mathbf{x}$, setting $\mathbf{F}$ and $\mathbf{H}$ to take canonical forms. We compare RTSNet to the MB KS, which achieves the MMSE lower bound when given access to perfect information. In Fig. 2 we consider a 2×2 system, where both algorithms are given access to an observation matrix $\mathbf{H}$ rotated by α degrees. We observe in Fig. 2 that when provided with the true $\mathbf{H} = \mathbf{H}_{\alpha=0^\circ}$, RTSNet, which does not know the noise statistics, coincides with the MB KS that knows the SS model, thus achieving the MMSE. Furthermore, when both algorithms are plugged in with a 10° rotated

[1]The source code along with additional information on the numerical study can be found online at `https://github.com/KalmanNet/RTSNet_ICASSP22`.

Table 1: Scaling and generalization, $T = 1000$, $\mathrm{r}^2 = 0\,[\mathrm{dB}]$.

System	2×2	5×5	10×10
T training	100	20	20
RTSNet MSE [dB]	-11.8204	-12.0535	-12.0766
MB KS MSE [dB]	-11.8689	-12.5480	-12.3985

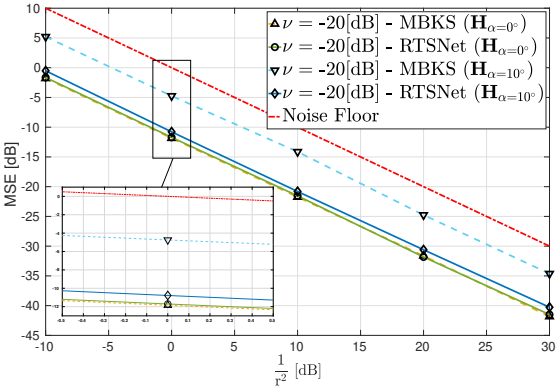


Fig. 2: MSE with and without mismatch, linear SS model.

matrix, RTSNet learns to apply alternative KGs, achieving an average gain of $5.98\,[\mathrm{dB}]$ over the MB KS, and within a minor gap of $1.04\,[\mathrm{dB}]$ from the MMSE. The results reported in Table 1, where RTSNet was trained on short T and tested on sequences of length $T = 1000$ for different system sizes, demonstrate that RTSNet can be trained to achieve the MMSE lower bound even for larger linear systems, and it does not overfit the training sequence length.

4.2. Lorenz Attractor

We evaluate RTSNet on the highly non-linear Lorenz attractor, a three-dimensional chaotic solution to the Lorenz ordinary differential equations, comparing it to the MB extended KS. The noiseless continuous-time state $\tilde{\mathbf{x}}_\tau$ evolves via

$$\frac{\partial}{\partial \tau}\tilde{\mathbf{x}}_\tau = \mathbf{A}\left(\tilde{\mathbf{x}}_\tau\right)\cdot\tilde{\mathbf{x}}_\tau, \quad \mathbf{A}\left(\mathbf{x}\right) = \begin{pmatrix} -10 & 10 & 0 \\ 28 & -1 & -x_1 \\ 0 & x_1 & -\frac{8}{3} \end{pmatrix}, \tag{6}$$

where $\tau \in \mathbb{R}^+$. The discrete-time model is approximated by the Jth order Taylor series (7) with sampling interval $\Delta\tau$; i.e.,

$$\mathbf{x}_{t+1} = \mathbf{F}\cdot\left(\mathbf{x}_t\right)\cdot\mathbf{x}_t, \quad \mathbf{F}\left(\mathbf{x}\right) \approx \sum_{j=0}^{J}\frac{\left(\mathbf{A}\left(\mathbf{x}\right)\cdot\Delta\tau\right)^j}{j!}. \tag{7}$$

For $\mathbf{h}(\cdot)$ we again use the canonical linear model.

In Fig. 3 we observe that given $\mathbf{H}_{\alpha=0^\circ}$, the extended MB KS and RTSNet achieve roughly the same MSE, while the latter does not require knowledge of the noise statistics. When the rotated $\mathbf{H}_{\alpha=1^\circ}$ is used, RTSNet learns to overcome such mismatches from data and to notably outperform the MB KS, which is sensitive to model uncertainty.

We conclude by evaluating RTSNet on long trajectories ($T = 3000$) with mismatches due to sampling a continuous-time process into discrete-time and comparing to a DD

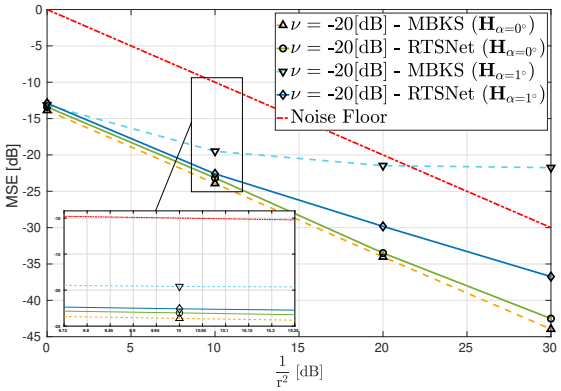


Fig. 3: MSE with and without mismatch, Lorenz attractor.

Table 2: Sampling and decimation.

Model	MB KS	Benchmark [19]	RTSNet
MSE [dB]	-10.071	-15.346	-15.56
Inference time [sec]	9.93	30.5	5.007
Training time [hours/epoch]	N/A	0.4	0.16
Number of trainable parameters	N/A	41, 236	33, 270

smoother benchmark [19]. Here, the data was generated at a very high time resolution and sub-sampled with a ratio of $\frac{1}{2000}$ to get a decimated process with $\Delta\tau_{\mathrm{d}} = 0.02$. No process noise was applied, and we set $\frac{1}{\mathrm{r}^2} = 0\,[\mathrm{dB}]$. The results reported in Table 2 demonstrate that RTSNet not only achieves the lowest MSE, but also does it most rapidly, with the smallest training overhead. Fig. 4 visualizes how this gain is clearly translated into improved tracking of the trajectory.

5. CONCLUSIONS

We presented RTSNet, a hybrid MB/DD implementation of the KS. RTSNet preserves the flow of the MB KS while learning the computation of the KGs using dedicated RNNs. The resulting architecture is shown to learn to implement the KS from data while overcoming model mismatches, and improving upon previously proposed DNN-based smoothers in terms of both performance and inference speed.

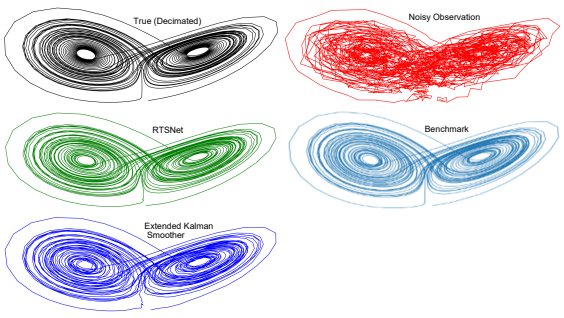


Fig. 4: Lorenz attractor with sampling mismatch, $T = 3000$.

6. REFERENCES

[1] J. Durbin and S. J. Koopman, *Time series analysis by state space methods*. Oxford University Press, 2012.

[2] N. Wiener, *Extrapolation, interpolation, and smoothing of stationary time series: With engineering applications*. MIT press Cambridge, MA, 1949, vol. 8.

[3] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[4] M. Gruber, "An approach to target tracking," MIT Lexington Lincoln Lab, Tech. Rep., 1967.

[5] S. J. Julier and J. K. Uhlmann, "New extension of the Kalman filter to nonlinear systems," in *Signal Processing, Sensor Fusion, and Target Recognition VI*, vol. 3068, 1997, pp. 182–193.

[6] H. E. Rauch, F. Tung, and C. T. Striebel, "Maximum likelihood estimates of linear dynamic systems," *AIAA Journal*, vol. 3, no. 8, pp. 1445–1450, 1965.

[7] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.

[8] Y. Bengio, "Learning deep architectures for AI," *Foundations and Trends® in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.

[9] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

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

[11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit *et al.*, "Attention is all you need," *arXiv preprint arXiv:1706.03762*, 2017.

[12] M. Zaheer, A. Ahmed, and A. J. Smola, "Latent LSTM allocation: Joint clustering and non-linear dynamic modeling of sequence data," in *International Conference on Machine Learning*, 2017, pp. 3967–3976.

[13] R. G. Krishnan, U. Shalit, and D. Sontag, "Deep Kalman filters," *preprint arXiv:1511.05121*, 2015.

[14] M. Karl, M. Soelch, J. Bayer, and P. Van der Smagt, "Deep variational Bayes filters: Unsupervised learning of state space models from raw data," *preprint arXiv:1605.06432*, 2016.

[15] M. Fraccaro, S. D. Kamronn, U. Paquet, and O. Winther, "A disentangled recognition and nonlinear dynamics model for unsupervised learning," in *Advances in Neural Information Processing Systems*, 2017.

[16] C. Naesseth, S. Linderman, R. Ranganath, and D. Blei, "Variational sequential Monte Carlo," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2018, pp. 968–977.

[17] E. Archer, I. M. Park, L. Buesing, J. Cunningham, and L. Paninski, "Black box variational inference for state space models," *arXiv preprint arXiv:1511.07367*, 2015.

[18] R. Krishnan, U. Shalit, and D. Sontag, "Structured inference networks for nonlinear state space models," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, 2017.

[19] V. G. Satorras, Z. Akata, and M. Welling, "Combining generative and discriminative models for hybrid inference," in *Advances in Neural Information Processing Systems*, 2019, pp. 13 802–13 812.

[20] N. Shlezinger, J. Whang, Y. C. Eldar, and A. G. Dimakis, "Model-based deep learning," *arXiv preprint arXiv:2012.08405*, 2020.

[21] N. Shlezinger, N. Farsad, Y. C. Eldar, and A. J. Goldsmith, "ViterbiNet: A deep learning based Viterbi algorithm for symbol detection," *IEEE Trans. Wireless Commun.*, vol. 19, no. 5, pp. 3319–3331, 2020.

[22] N. Shlezinger, R. Fu, and Y. C. Eldar, "DeepSIC: Deep soft interference cancellation for multiuser MIMO detection," *IEEE Trans. Wireless Commun.*, vol. 20, no. 2, pp. 1349–1362, 2021.

[23] N. Shlezinger, N. Farsad, Y. C. Eldar, and A. J. Goldsmith, "Learned factor graphs for inference from stationary time sequences," *arXiv preprint arXiv:2006.03258*, 2020.

[24] A. López Escoriza, G. Revach, N. Shlezinger, and R. J. G. van Sloun, "Data-driven Kalman-based velocity estimation for autonomous racing," in *Proc. IEEE ICAS*, 2021.

[25] G. Revach, N. Shlezinger, X. Ni, A. L. Escoriza, R. J. van Sloun, and Y. C. Eldar, "KalmanNet: Neural network aided Kalman filtering for partially known dynamics," *arXiv preprint arXiv:2107.10043*, 2021.

[26] G. J. Bierman, *Factorization methods for discrete sequential estimation*. Courier Corporation, 1977.

[27] F. Wadehn, *State space methods with applications in biomedical signal processing*. ETH Zurich, 2019, vol. 31.