RTSNet - Data Driven Kalman Smoothing

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

ICASSP 2022

Motivation

Tracking of dyanmic systems is encountered in many applications:

- Localization
- Navigation
- Task Planning

such settings can often be represented as smoothing tasks, which are typically tackled using either a Model-Based(MB) or a Data-Driven(DD) method.

Model-based Deep Learning

In this work we aim to design a hybrid MB DD smoother.

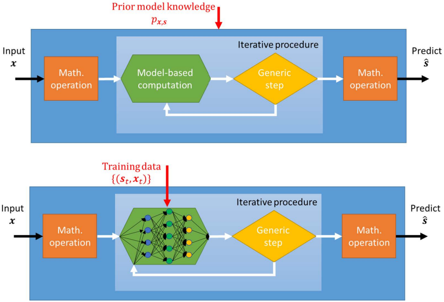


Figure: DNN-aided inference illustration[1]

Key idea: replace part of the MB computation by NN, in order to incorporate the advantages of both domains.

[1] Nir Shlezinger et al. "Model-Based Deep Learning". In: *arXiv preprint arXiv:2012.08405* (2020)

Agenda

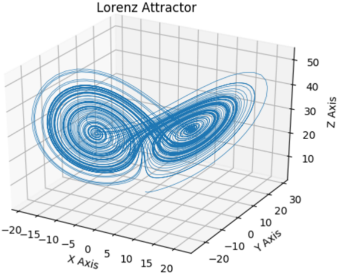
- Smoothing Problem Formulation
- RTSNet Architecture
- Experiments on Linear and Non-linear Cases

Smoothing Problem Formulation

Consider *fixed-interval* smoothing: 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

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

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



Traditional Model-Based Solution

Linear case:

Rauch-Tung-Striebel (RTS) Smoother achieves the optimal MMSE for linear State Space model ☺

Non-linear case:

- Extended RTS smoother
  - Subject to Linearization error ☹
- Particle smoother
  - performance is unstable and hard to quantify ☹
  - computation complexity increases dramatically with the number of particles ☹

These drawbacks motivate deriving a **NN**-aided Kalman Smoother.

RTS Smoother Review

The MB RTS Smoother recovers the latent state variables using the forward and backward passes.

The forward pass is a standard Kalman Filter (KF), Where $\mathcal{K}_t$ is the *forward* Kalman Gain (KG):

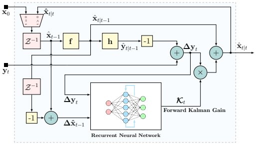
$$\mathcal{K}_t = \hat{\Sigma}_{t|t-1} \cdot \hat{\mathbf{H}}^\top \cdot \hat{\mathbf{S}}_t^{-1}. \tag{2}$$

On the other hand, the *backward* KG $\mathcal{G}_t$ is given by,

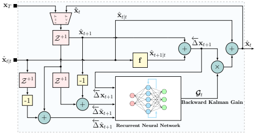
$$\mathcal{G}_t = \hat{\Sigma}_{t|t} \cdot \hat{\mathbf{F}}^\top \cdot \hat{\Sigma}_{t+1|t}^{-1}. \tag{3}$$

all domain knowledge encapsulated in KGs.

RTSNet Architecture



(a) Forward pass.



(b) Backward pass.

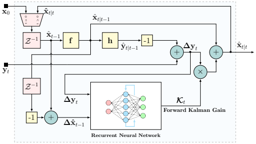
- Choose RTS as Backbone: all domain knowledge encapsulated in KGs.

$$\mathcal{K}_t = \hat{\Sigma}_{t|t-1} \cdot \hat{\mathbf{H}}^\top \cdot \hat{\mathbf{S}}_t^{-1}. \quad (4)$$

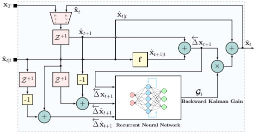
$$\mathcal{G}_t = \hat{\Sigma}_{t|t} \cdot \hat{\mathbf{F}}^\top \cdot \hat{\Sigma}_{t+1|t}^{-1}. \quad (5)$$

- Replace *forward* KG (4) and *backward* KG (5) with **NN**s, where Low- complexity **NN** consists of an input FC, a two-layer GRU and an output FC layer.

Architecture Discussion



(a) Forward pass.

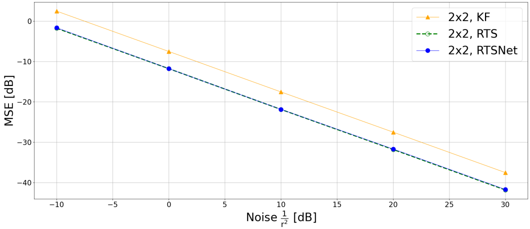


(b) Backward pass.

- NN-aided KGs compensate for model mismatch
- Avoid linearization and is less sensitive to non-linearities
- **Not** require inverting matrices while inferring rapidly with low computation complexity due to efficient RNNs
- Utilize a single learned forward-backward pass, which can be extended to carry out multiple passes via deep unfolding

Linear Model

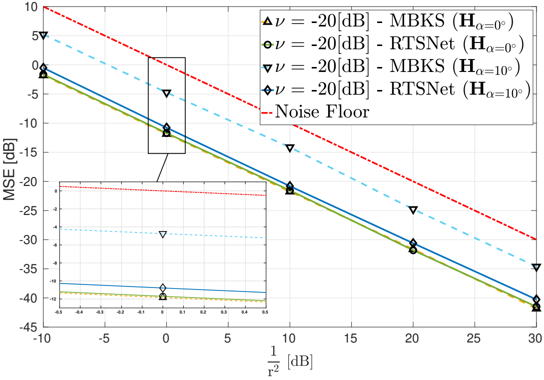
- For *Linear* State-space Model with *Gaussian* noise, RTS smoother is optimal.
- Synthetic linear dataset: set F and H to take the controllable canonical and inverse canonical forms, respectively.



- Our **RTSNet converges to the optimal** RTS smoother.

Linear - Model Mismatch

Rotate observation matrix H by 10°.



Similar results can be achieved when rotate F.

RTSNet is superior to RTS smoother for model mismatch.

Linear - Generalization

- Scale SS model F & H to 10x10
- Scale T_{test} to 1000

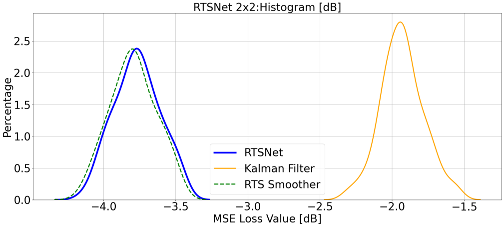


Figure: Training trajectory length 20, testing trajectory length 1000

	KF	RTS	RTSNet
MSE Loss [dB]	-1.9271	-3.7917	-3.7658

Lorenz Attractor - Sampling and decimation

Evaluate RTSNet on long trajectories (T = 3000) with mismatches due to sampling a continuous-time process into discrete-time.

Compare with DD Benchmark: Similar MSE performance, much better training time and inference time.

Table: Sampling and decimation.

Model	MB KS	Benchmark[2]	RTSNet
mean-squared error (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	$41,236$	$33,270$

[2] Victor Garcia Satorras, Zeynep Akata, and Max Welling. "Combining Generative and Discriminative Models for Hybrid Inference". In: *Advances in Neural Information Processing Systems*. 2019, pp. 13802–13812

Lorenz Attractor - Sampling and decimation - Trajectories

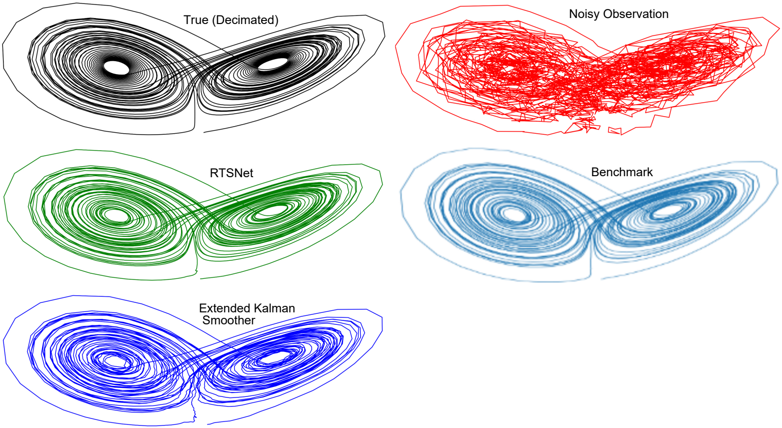
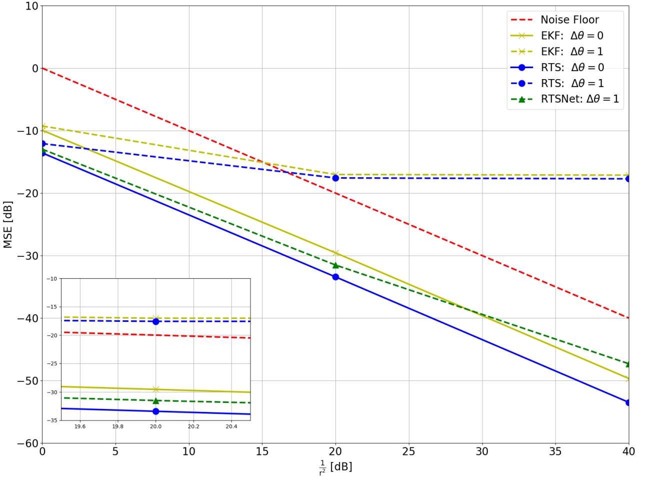


Figure: Lorenz attractor with sampling mismatch, $T = 3000$.

Lorenz Attractor - Model Mismatch



Future Work

1. Evaluate RTSNet on real-world data-set, e.g, NCLT.
2. Extend the network to handle jumps in the hidden state and to detect outlier observations, possibly using NUV priors.
3. Try fixed-lag smoothing with sliding window. (Although fix-lag can face computation inefficiency problem, it is sometimes of more practical use.)
4. Enable RTSNet to face asynchronous mearsurement update.

Check Us

Check us on Arxiv

Check us on GitHub

References

- Satorras, Victor Garcia, Zeynep Akata, and Max Welling. "Combining Generative and Discriminative Models for Hybrid Inference". In: *Advances in Neural Information Processing Systems*. 2019, pp. 13802–13812.
- Shlezinger, Nir et al. "Model-Based Deep Learning". In: *arXiv preprint arXiv:2012.08405* (2020).