南開大學
Nankai University

Deep learning based state estimation

Incorporating Transformer and LSTM to Kalman Filter with EM algorithm

Z. Shi

Nankai University

January 7, 2021

Overview

- Kalman Filter requires the true parameters of the model and solves optimal state estimation recursively. Expectation Maximization (EM) algorithm is applicable for estimating the parameters of the model that are not available before Kalman filtering, which is EM-KF algorithm.
- To improve the preciseness of EM-KF algorithm, the author presents a state estimation method by combining the Long-Short Term Memory network (LSTM), Transformer and EM-KF algorithm in the framework of Encoder-Decoder in Sequence to Sequence (seq2seq).
- Simulation on a linear mobile robot model demonstrates that the new method is more accurate.

Kalman Filter

- As systems are usually interfered by stochastic noise, state estimation aims to estimate the true state via observation, and minimize the error between estimation and true state.
- R. E. Kalman proposed Kalman Filter (KF) [1] to recursively estimate state via observation in stochastic linear systems.

$$\begin{aligned} x_k &= Ax_{k-1} + w_k, \\ y_k &= Cx_k + v_k, \\ w_k &\sim N(0, Q), v_k \sim N(0, R), \\ x_0 &\sim N(m_0, P_0),\ k = 1, 2, ..., N, \end{aligned} \tag{1}$$

- Nonlinear extension: Extended Kalman Filter (EKF), Unscented Kalman Filter (UKF, also known as sigma-point Kalman Filter)

Disadvantages of KF

1. KF requires model parameters. Although A, C are easy to obtain through system modeling, Q, R, m_0, P_0 depend on estimation by designers' experience. Solution: incorporating Expectation Maximization (EM) algorithm [2] for parameter estimation
2. Eq. (1) clarifies that KF assumpts Markov property of states, and conditional independence of observations, yet de facto systems do not follow these two assumptions usually. Solution: deep learning

Related works

- EM based KF (Shumway and Stoffer,1982)[3]⇒ (Ghahramani and Hinton,1996)[4] ⇒ nonlinear extensions
- Deep learning for state estimation
  - State estimation is a kind of time-series forecasting.
  - Recurrent Neural Network (RNN) [5], Long Short-Term Memory (LSTM) [6]
  - seq2seq [7]: Encoder-decoder
  - Attention Mechanism [8], Transformer [9]

Framework

- We proposed encoder-decoder framework in seq2seq for state estimation, that state estimation is equivalent to encode and decode observation.
  1. Previous works incorporating LSTM to KF, are adopting LSTM encoder and KF decoder. We proposed LSTM-KF adopting LSTM encoder and EM-KF decoder.
  2. Before EM-KF decoder, replace LSTM encoder by Transformer encoder, we call this Transformer-KF.
  3. Integrating Transformer and LSTM, we call this TL-KF.
- Integrating Transformer and LSTM to encode observation before filtering, makes it easier for EM algorithm to estimate parameters.

KF and EM

- Kalman Filter (KF) $x_{k-1} \to x_k$, forward recursively.

$$p(x_k|y_{1:k}) = N(x_k|m_{k|k}, P_{k|k}). \tag{2}$$

- Kalman Smoother (KS) [10], based on KF, backward recursively. (See [11] for details)

$$p(\hat{x}_k|y_{1:N}) = N(\hat{x}_k|m_{k|N}, P_{k|N}), \tag{3}$$

- Expectation Maximization (EM) algorithm obtains maximum likelihood estimation of parameters iteratively, with latent variables in probability distribution function. State estimation maximize loglikelihood of observation, which is equivalent to maximize following expectation.

$$\mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{(n)}\right) = \mathbb{E}\left[\ln p\left(\mathbf{x}_{0:N}, \mathbf{y}_{1:N}|\boldsymbol{\theta}\right)|\mathbf{y}_{1:N}, \boldsymbol{\theta}^{(n)}\right]. \tag{4}$$

EM-KF algorithm

$$
\begin{aligned}
\mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{(n)}\right) = & -\frac{1}{2}\ln\left|2\pi\mathbf{P}_0\right| - \frac{N}{2}\ln\left|2\pi\mathbf{Q}\right| - \frac{N}{2}\ln\left|2\pi\mathbf{R}\right| \\
& -\frac{1}{2}\mathrm{tr}\left\{\mathbf{P}_0^{-1}\left[\mathbf{P}_{0|N} + \left(\mathbf{m}_{0|N} - \mathbf{m}_0\right)\left(\mathbf{m}_{0|N} - \mathbf{m}_0\right)^T\right]\right\} \\
& -\frac{1}{2}\sum_{k=1}^{N}\mathrm{tr}\left\{\mathbf{Q}^{-1}\mathbb{E}\left[\left(\mathbf{x}_k - \mathbf{A}\mathbf{x}_{k-1}\right)\left(\mathbf{x}_k - \mathbf{A}\mathbf{x}_{k-1}\right)^T\left|\mathbf{y}_{1:N}\right.\right]\right\} \\
& -\frac{1}{2}\sum_{k=1}^{N}\mathrm{tr}\left\{\mathbf{R}^{-1}\mathbb{E}\left[\left(\mathbf{y}_k - \mathbf{C}\mathbf{x}_k\right)\left(\mathbf{y}_k - \mathbf{C}\mathbf{x}_k\right)^T\left|\mathbf{y}_{1:N}\right.\right]\right\}.
\end{aligned}
\tag{5}
$$

EM-KF algorithm

Algorithm 1 EM-KF algorithm

Require: Initial estimation of parameters $\boldsymbol{\theta}^{(0)}$, error ϵ, maximum iteration n_m
Ensure: $\boldsymbol{\theta}^*$

1: **repeat**
2: E-step: compute $\mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{(n)}\right)$
3: M-step: $\boldsymbol{\theta}^{(n+1)} \leftarrow \arg\max_{\boldsymbol{\theta}} \mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{(n)}\right)$ via $\frac{\partial \mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{(n)}\right)}{\partial \boldsymbol{\theta}^{(n)}} = 0$
4: **until** $\left|\boldsymbol{\theta}^{(n+1)} - \boldsymbol{\theta}^{(n)}\right| < \epsilon$, or up to maximum iteration
5: **return** $\boldsymbol{\theta}^* \leftarrow \boldsymbol{\theta}^{(n+1)}$

Why deep learning for state estimation?

- Connection among seq2seq, hidden Markov model (HMM), and KF in linear dynamic system (LDS)
- Since EM-KF depends on observation only, it may not estimate parameters w.r.t. state Q, m_0, P_0 precisely, it is only competent to estimate R. Observation can not depict information of state
- Deep learning for observation is capable of mining more information w.r.t. state, to enhance the performance.

Combining LSTM, Transformer and EM-KF

- **LSTM-KF** Input observation into LSTM, output a new series that depict state more effectively. Despite the difference of R between new series and old one, as EM-KF can estimate R, we only need to replace old series by this new one to estimate Q, m_0, P_0 precisely.
- **Transformer-KF** Transformer can capture long-term dependency (that LSTM may not) [12], which enhance the robustness for observation with noise. Besides, Transformer is an encoder-decoder model, which is easy to incorporate into our proposed encoder-decoder framework.
- **TL-KF**It is not sufficient to depict time-series by position encoding in Transformer. Instead, we can adopt a Transformer-LSTM structure.

Why first Transformer, then LSTM?

1. Attention is usually before memory in human cognitive system.
2. The reason why Transformer can capture long-term dependency, is that it integrates multi-head self attention and residual [13] connection [14], and position encoding in Transformer need to be improved [15]. Combining LSTM and Transformer can enhance both structural advantages and ability for time-series modeling of Transformer.
3. Transformer draws on designs of convolutional neural networks. (Eg. multi-head attention vs. multi convolutional kernel, residual connection etc.) Therefore, Transformer can capture saliency that RNN may not, while RNN can better depict time-series.

Deep learning based state estimation

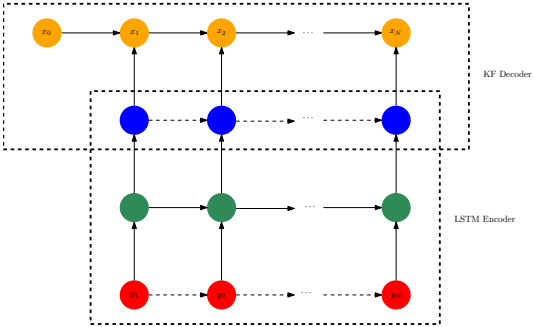


Figure 1: LSTM-KF, with LSTM encoder and KF decoder, where dashed lines denote latent connections.

Deep learning based state estimation

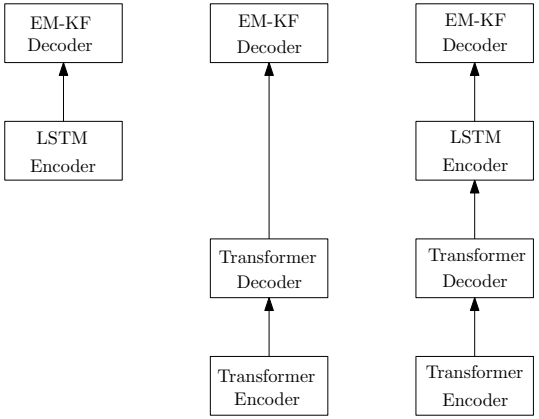


Figure 2: Deep learning models for state estimation

Initial settings I

- Linear mobile robot model on one DOF, with displacement, velocity and accerelation as state variables. Displacement is observed by Efficient Perspective-n-Points (EPnP) algorithm. [16, 11]

$$\begin{pmatrix} x_k \\ v_k \\ a_k \end{pmatrix} = \begin{pmatrix} 1 & T & 0.5T^2 \\ 0 & 1 & T \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_{k-1} \\ v_{k-1} \\ a_{k-1} \end{pmatrix} + w_k, \tag{6}$$
$$y_k = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix} x_k + v_k,$$
$$w_k \sim N(0, Q), v_k \sim N(0, R), x_0 \sim N(m_0, P_0).$$

- Q, R, m_0, P_0 unknown, first adopt EM algorithm for estimating R, then generate new series for estimating Q, m_0, P_0, to compare EM-KF, LSTM-KF, Transformer-KF and TL-KF.
- Suppose Q, R, P_0 are as form $\sigma^2 I$, and $m_0$is as form $(0, 0, m_a)^T$.

Initial settings II

- σ^2 estimation = trace of covariance matrix / order of matrix
- m_{03} is the estimation of m_a.

- EM iteration 10 times
- de facto values $Q = 1 \times 10^{-2} I_3, R = 5 \times 10^{-3} I_1, m_0 = (0, 0, 0.1)^T, P_0 = 0.1 I_3$, sampling term $T = 0.01$ second, sequence length $N = 200$
- initial set $Q = 2 \times 10^{-2} I_3, R = I_1, m_0 = (0, 0, 1)^T, P_0 = 5 I_3$.
- Python 3.7.1 + Anaconda 5.3.1 + Pytorch 1.0 (64-bit, no CUDA).

Tables I

Table 1: Comparison on parameters estimation. (1) EM-KF estimates R accurately. (2) Transformer-KF estimates Q accurately. (3) LSTM-KF estimates m_0, P_0 accurately. (4) TL-KF estimates Q, m_0, P_0 accurately.

	σ_q^2	σ_r^2	m_a	σ_p^2
de facto values	1.0×10^{-2}	5.0×10^{-3}	0.1	0.1
EM-KF	2.04×10^{-2}	4.23×10^{-3}	0.879	0.417
LSTM-KF	0.14×10^{-2}	34.3×10^{-3}	0.109	0.116
Transformer-KF	1.18×10^{-2}	98.4×10^{-3}	0.768	0.359
TL-KF	1.27×10^{-2}	44.8×10^{-3}	0.121	0.118

Tables II

Table 2: Change the initial setting of m_0, P_0, then estimate parameters. TL-KF is more robust than EM-KF.

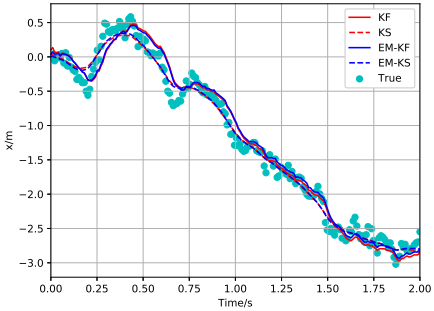
Method	initial m_a	initial σ_p^2	estimating σ_q^2	estimating m_a	estimating σ_p^2
EM	1	5	0.0204	0.879	0.417
EM	0.5	1	0.3447	0.794	0.239
EM	1.5	15	0.0255	1.195	0.403
TL	1	5	0.0126	0.121	0.118
TL	0.5	1	0.0123	0.091	0.097
TL	1.5	15	0.0128	0.089	0.120

Tables III

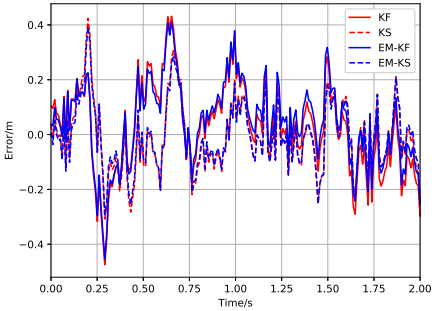
Table 3: Comparison on state estimation

	Training time (sec)	EM time (sec)	Filter MSE(m^2)	Smoother MSE(m^2)
KF			26.83×10^{-3}	16.95×10^{-3}
EM-KF		3.98	26.32×10^{-3}	15.19×10^{-3}
LSTM-KF	76.82	4.15	17.51×10^{-3}	11.52×10^{-3}
Transformer-KF	26.71	4.22	9.05×10^{-3}	6.55×10^{-3}
TL-KF	91.46	4.23	5.07×10^{-3}	4.89×10^{-3}

Figures I



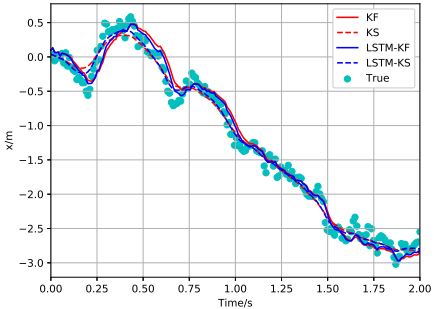
(a) path



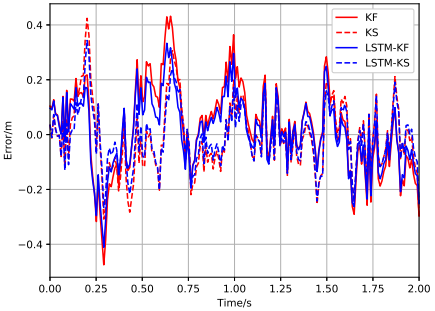
(b) error

Figure 3: EM-KF

Figures II



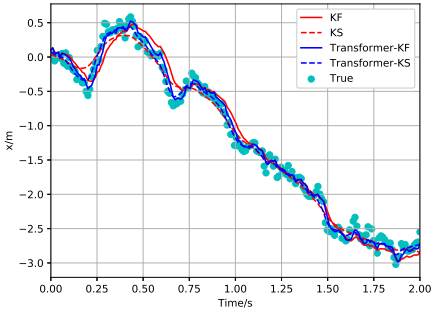
(a) path



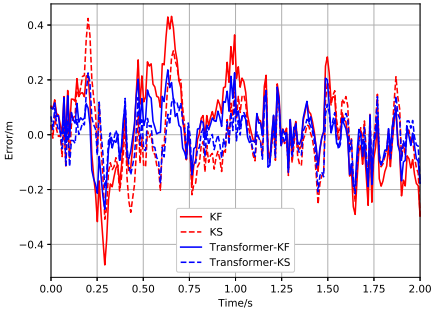
(b) error

Figure 4: LSTM-KF

Figures III



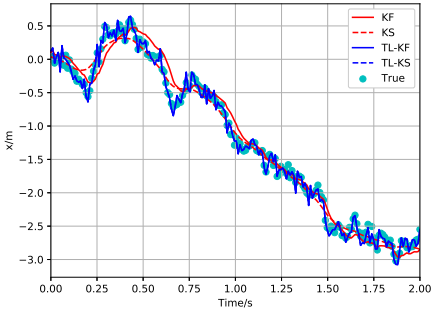
(a) path



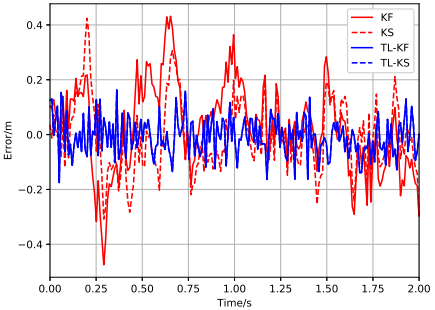
(b) error

Figure 5: Transformer-KF

Figures IV



(a) path



(b) error

Figure 6: TL-KF

Conclusions

1. Combining Transformer and LSTM as an encoder-decoder framework for observation, can depict state more effectively, attenuate noise interference, and weaken the assumption of Markov property of states, and conditional independence of observations. This can enhance the preciseness and robustness of state estimation.
2. Transformer, based on multi-head self attention and residual connection, can capture long-term dependency, while LSTM-encoder can model time-series. TL-KF, a combination of Transformer, LSTM and EM-KF, is precise for state estimation in systems with unknown parameters.
3. Kalman smoother can ameliorate Kalman filter, but in TL-KF, filtering is precise enough. Therefore, after offline training for estimating Q, m_0, P_0, KF for online estimation can be adopted.

References I

[1] R. E. Kalman. "A new approach to linear filtering and prediction problems". In: *Journal of Fluids Engineering* 82.1 (1960), pp. 35–44.

[2] A. P. Dempster, N. M. Laird, and D. B. Rubin. "Maximum likelihood from incomplete data via the EM algorithm". In: *Journal of the Royal Statistical Society* 39.1 (1977), pp. 1–38.

[3] R. H. Shumway and D. S. Stoffer. "An approach to time series smoothing and forecasting using the EM algorithm". In: *Journal of Time Series Analysis* 3.4 (1982), pp. 253–264.

[4] Zoubin Ghahramani and Geoffrey Hinton. *Parameter estimation for linear dynamical systems*. Feb. 22, 1996. URL: http://mlg.eng.cam.ac.uk/zoubin/papers/tr-96-2.pdf (visited on 05/23/2019).

References II

[5] Jeffrey L. Elman. "Finding structure in time". In: *Cognitive Science* 14.2 (1990), pp. 179–211.

[6] Alex Graves. "Long short-term memory". In: *Neural Computation* 9.8 (1997), pp. 1735–1780.

[7] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. "Sequence to sequence learning with neural networks". In: *NeurIPS*. 2014.

[8] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate". In: *ICLR*. 2015.

[9] Ashish Vaswani et al. "Attention is all you need". In: *NeurIPS*. 2017.

[10] RAUCH et al. "Maximum likelihood estimates of linear dynamic systems". In: *AIAA Student Journal American Institute of Aeronautics and Astronautics* 3.8 (1965), pp. 1445–1450.

References III

[11] Timothy Barfoot. *State estimation for robotics*. Cambridge University Press, 2017.

[12] T. Hollis, A. Viscardi, and S.E. Yi. "A comparison of LSTMs and attention mechanisms for forecasting financial time series". In: *CoRR* (2018). arXiv: 1812.07699.

[13] Kaiming He et al. "Deep residual learning for image recognition". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 770–778.

[14] T. Domhan. "How much attention do you need? a granular analysis of neural machine translation architectures". In: *Association for Computational Linguistics*. 2018.

[15] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. "Self-attention with relative position representations". In: *NAACL*. 2018.

References IV

[16] V. LEPETIT, N. F. MORENO, and P. FUA. “Epnp: an accurate o (n) solution to the pnp problem”. In: *International Journal of Computer Vision* 81.2 (2009), pp. 155–166.