Optimization or Architecture: What Matters in Non-Linear Filtering?

Ido Greenberg [1] Netanel Yannay Shie Mannor [1 2]

Abstract

In non-linear filtering, it is traditional to compare non-linear architectures such as neural networks to the standard linear Kalman Filter (KF). We observe that this methodology mixes the evaluation of two separate components: the non-linear architecture, and the numeric optimization method. In particular, the non-linear model is often optimized, whereas the reference KF model is not. We argue that *both* should be optimized similarly. We suggest the Optimized KF (**OKF**), which adjusts numeric optimization to the positive-definite KF parameters. We demonstrate how a significant advantage of a neural network over the KF may *entirely vanish* once the KF is optimized using OKF. This implies that experimental conclusions of certain previous studies were derived from a flawed process. The benefits of OKF over the non-optimized KF are further studied theoretically and empirically, where OKF consistently improves the accuracy in a variety of problems. Experiments are available on Github, and the OKF on PyPI.

1. Introduction

The Kalman Filter (KF) (Kalman, 1960) is a celebrated method for linear filtering, with applications in tracking, guidance, navigation, control and other fields (Zarchan and Musoff, 2000; Kirubarajan, 2002). The KF provides optimal predictions under certain assumptions (namely, linear models with i.i.d noise). In practical problems, these assumptions are often violated, rendering the KF sub-optimal and motivating the growing field of non-linear filtering. Many studies demonstrated the benefits of non-linear models over the KF (Revach et al., 2022; Coskun et al., 2017).

Originally, we sought to join this line of works. Motivated by a real-world radar problem, we developed a dedicated Neural KF (NKF) based on the LSTM sequential model. NKF achieved significantly better accuracy than the KF. Then, during ablation tests, we noticed that the KF and NKF differ in *both architecture and optimization*. Specifically, the KF depends on the noise parameters, which are traditionally determined by noise estimation (Odelson et al., 2006); whereas NKF's parameters are optimized using supervised learning methods. To fairly evaluate the two architectures, we wished to apply the same optimization to both. To that end, we devised an Optimized KF (**OKF**, Section 3). KF and OKF have the same linear architecture: OKF only changes the noise parameters values. Yet, OKF *outperformed* NKF, reversed the whole experimental conclusion, and made NKF unnecessary for this problem (Section 4).

Our original erroneous methodology compared two different models (KF and NKF) that were not optimized similarly. A review of the non-linear filtering literature reveals that this methodology is used in many studies. Specifically, for a baseline KF model, the parameters are often tuned by noise estimation (fa Dai et al., 2020; Aydogmus and Aydogmus, 2015; Revach et al., 2022); by heuristics (Jamil et al., 2020; Coskun et al., 2017; Ullah et al., 2019); or are simply ignored (Gao et al., 2019; Bai et al., 2020; Zheng et al., 2019), often without public code for examination. In all these studies, the complex, non-linear model *may* be beneficial; however, this conclusion cannot be inferred from the experimental evidence. Hussein (2014) even discusses the (Extended-)KF sensitivity to its parameters, and suggests a neural network with supervised learning – yet never considers the same supervised learning for the KF itself.

So far, OKF is presented as a ***methodological* contribution for non-linear filtering**: it is closer than KF to standard non-linear methods, and can be used as a reliable baseline for comparison. In addition, OKF also provides a ***practical* contribution for linear filtering**: it outperforms the KF, and can be used for more accurate filtering. This is demonstrated extensively in Appendix B and Appendix C – in different domains, over different problem variations, with different data sizes, and even under distributional shifts.

Notice that OKF outperforms KF using the *same* linear architecture. This may come as a surprise: for this architecture, the standard KF tuning method is known to be already optimal. However, this optimality depends on restrictive assumptions which are commonly violated, leaving room for optimization by OKF. Appendix B analyzes two violations –

[1] Technion, Israel [2] Nvidia Research. Correspondence to: Ido Greenberg <gido@campus.technion.ac.il>.

Accepted after peer-review at the SCIS workshop, ICML, Honolulu, Hawaii, USA. July, 2023. Copyright 2023 by the author(s).

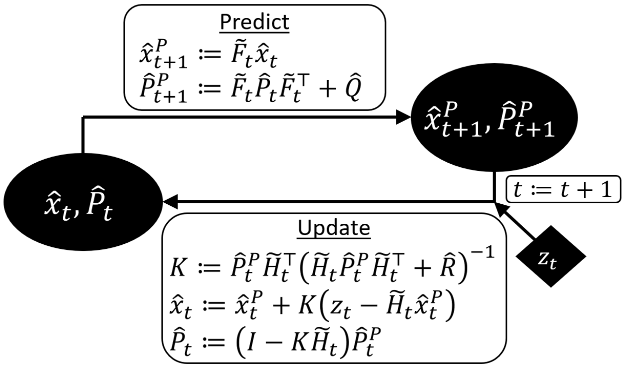

Predict
$\hat{x}_{t+1}^P \coloneqq \tilde{F}_t \hat{x}_t$
$\hat{P}_{t+1}^P \coloneqq \tilde{F}_t \hat{P}_t \tilde{F}_t^\top + \hat{Q}$
$\hat{x}_{t+1}^P, \hat{P}_{t+1}^P$
$t \coloneqq t+1$
$\hat{x}_t, \hat{P}_t$
z_t
Update
$K \coloneqq \hat{P}_t^P \tilde{H}_t^\top \left(\tilde{H}_t \hat{P}_t^P \tilde{H}_t^\top + \hat{R}\right)^{-1}$
$\hat{x}_t \coloneqq \hat{x}_t^P + K\left(z_t - \tilde{H}_t \hat{x}_t^P\right)$
$\hat{P}_t \coloneqq \left(I - K\tilde{H}_t\right)\hat{P}_t^P$

Figure 1: The KF algorithm. The prediction step is based on the motion model $\tilde{F}_t$ with noise $\hat{Q}$, whereas the update step is based on the observation model $\tilde{H}_t$ with noise $\hat{R}$.

non-linear dynamics and non-i.i.d noise – and theoretically explains the KF's sub-optimality. Disturbingly, such violations may even cause the KF errors to deteriorate with the train data size (Appendix C.2).

Scope: We focus on the supervised filtering setting, where training data includes both observations and the true system states (whose prediction is usually the objective). Such data is available in many practical applications. For example, the states may be provided by external accurate sensors such as GPS fa Dai et al. (2020); by manual object labeling in computer vision (Wojke et al., 2017); by controlled experiments of radar targets; or by simulations of known dynamic systems. As demonstrated in the studies cited above, this supervised setting is common in non-linear filtering.

Contribution: (a) We point to a common methodological error – comparing an optimized filtering model to a non-optimized KF. (b) We introduce the Optimized KF (OKF) as a remedy: as demonstrated, using OKF as a baseline for comparison may reverse the entire experimental conclusion. OKF is further motivated as a filtering algorithm, by (c) theoretical analysis of the KF sub-optimality, and (d) extensive demonstration of the superior accuracy of OKF over the KF.

2. Preliminaries

Consider the KF model for a dynamic system with no control signal (Kalman, 1960):

$$\begin{aligned} X_{t+1} &= F_t X_t + \omega_t \qquad (\omega_t \sim \mathcal{N}(0,Q)) \\ Z_t &= H_t X_t + \nu_t \qquad (\nu_t \sim \mathcal{N}(0,R)). \end{aligned} \tag{1}$$

X_t is the system state at time t, and its estimation is usually the goal. Its dynamics are modeled by the linear operator F_t, with random noise ω_t whose covariance is Q. Z_t is the observation, modeled by the operator H_t with noise ν_t whose covariance is R. The notation may be simplified to F, H in the stationary case.

The KF represents X_t via estimation of the mean $\hat{x}_t$ and covariance $\hat{P}_t$. As shown in Fig. 1, the KF alternately predicts the next state (*prediction* step), and processes new information from incoming observations (*update* or *filtering* step). The KF relies on the matrices $\tilde{F}_t, \tilde{H}_t, \hat{Q}, \hat{R}$, intended to represent F_t, H_t, Q, R of Eq. (1). Whenever F_t, H_t are known and stationary, we may simplify the notation to $\tilde{F}_t = F$, $\tilde{H}_t = H$.

The KF estimator $\hat{x}_t$ is optimal in terms of mean square errors (MSE) – but only under a restrictive set of assumptions (Kalman, 1960):

Assumption 1 (KF assumptions)**.** $\tilde{F}_t = F_t$, $\tilde{H}_t = H_t$ are known and independent of X_t (*linear models*); each sequence $\{\omega_t\}, \{\nu_t\}$ is i.i.d; the covariances $\hat{Q} = Q$, $\hat{R} = R$ are known; and $\hat{x}_0, \hat{P}_0$ correspond to the mean and covariance of the initial X_0.

Theorem 1 (KF optimality; e.g., see Jazwinski (2007); Humpherys et al. (2012))**.** Under Assumption 1, the KF estimator $\hat{x}_t$ minimizes the MSE w.r.t. X_t.

The KF accuracy strongly depends on its parameters $\hat{Q}$ and $\hat{R}$ (Formentin and Bittanti, 2014). As motivated by Theorem 1, these parameters are usually identified with the noise covariance Q, R and are set accordingly: "*the systematic and preferable approach to determine the filter gain is to estimate the covariances from data*" (Odelson et al., 2006). In absence of system state data $\{x_t\}$ (the "ground truth"), many methods were suggested to estimate the covariances from observations $\{z_t\}$ alone (Mehra, 1970; Zanni et al., 2017; Park et al., 2019; Feng et al., 2014). We focus on the supervised setting, where the states $\{x_t\}$ are available in the training data (but not in inference).

Definition 1 (Supervised data)**.** Consider K trajectories of a dynamic system, with lengths $\{T_k\}_{k=1}^K$. We define their supervised data as the sequences of true system states $x_{k,t} \in \mathbb{R}^{d_x}$ and observations $z_{k,t} \in \mathbb{R}^{d_z}$: $\{\{(x_{k,t}, z_{k,t})\}_{t=1}^{T_k}\}_{k=1}^K$.

If F_t, H_t are known, the supervised setting permits a direct calculation of the sample covariance matrices of the noise (Lacey, 1998):

$$\begin{aligned} \hat{Q} &\coloneqq Cov(\{x_{k,t+1} - F_t x_{k,t}\}_{k,t}) \\ \hat{R} &\coloneqq Cov(\{z_{k,t} - H_t x_{k,t}\}_{k,t}). \end{aligned} \tag{2}$$

Since Theorem 1 guarantees optimality when $\hat{Q} = Q, \hat{R} = R$, and Eq. (2) provides a simple estimator for Q and R, Algorithm 1 has become the gold-standard tuning method for KF from supervised data.

Algorithm 1 (KF noise estimation)**.** Given supervised data $\{(x_{k,t}, z_{k,t})\}$, return $\hat{Q}$ and $\hat{R}$ of Eq. (2).

While Algorithm 1 is indeed trivial to apply in the supervised setting, we show below that when Assumption 1 is violated, it no longer provides optimal predictions.

3. Optimized Kalman Filter

In this section, we propose to determine the noise parameters $\hat{Q}, \hat{R}$ of Fig. 1 by optimizing the filtering errors directly – instead of estimating Q, R by Algorithm 1. This might seem equivalent: according to Theorem 1, $\hat{Q} = Q$, $\hat{R} = R$ already minimize the MSE! For this reason, as mentioned above, Algorithm 1 is indeed preferred in the supervised settings (Odelson et al., 2006), and other methods are only left for settings without supervised data (where Algorithm 1 cannot be applied).

However, the equivalence between MSE minimization and Algorithm 1 only holds under Assumption 1. As demonstrated below, Assumption 1 is violated in many problems – from real-world applications to simple toy problems – often in ways that are hard to even notice. This motivates replacing Algorithm 1 with explicit optimization – even if supervised data is available.

Formally, we consider the KF (Fig. 1) as a prediction model $\hat{x}_{k,t}(\{z_{k,\tau}\}_{\tau=1}^{t};\, \hat{Q}, \hat{R})$, which estimates $x_{k,t}$ given the observations $\{z_{k,\tau}\}_{\tau=1}^{t}$ and parameters $\hat{Q}, \hat{R}$. We define the KF optimization problem:

$$\begin{aligned}\operatorname*{argmin}_{Q',R'} \sum_{k=1}^{K}\sum_{t=1}^{T_k} \text{loss}\left(\hat{x}_{k,t}\left(\{z_{k,\tau}\}_{\tau=1}^{t};\, Q', R'\right),\ x_{k,t}\right) \\ \text{s.t. } Q' \in S_{++}^{d_x},\ R' \in S_{++}^{d_z},\end{aligned} \tag{3}$$

where $S_{++}^{d} \subset \mathbb{R}^{d\times d}$ is the space of Symmetric and Positive Definite matrices (SPD), and loss is the objective function (e.g., $\text{loss}(\hat{x}, x) = ||\hat{x} - x||^2$ for MSE). Note that the estimation problem may be modified to prediction, by changing the observed input from $\{z_{k,\tau}\}_{\tau=1}^{t}$ to $\{z_{k,\tau}\}_{\tau=1}^{t-1}$.

A significant challenge in solving Eq. (3) is the SPD constraint. While standard numeric optimization methods (e.g., Adam (Diederik P. Kingma, 2014)) can optimize sequential prediction models, they may violate the constraint. In other settings, the SPD constraint is often bypassed using diagonal restriction: "*since both the covariance matrices must be constrained to be positive semi-definite, Q and R are often parameterized as diagonal matrices*" (Formentin and Bittanti, 2014). To maintain the complete expressiveness of $\hat{Q}$ and $\hat{R}$ throughout the optimization, we instead use the Cholesky parameterization (Pinheiro and Bates, 1996).

The parameterization relies on Cholesky decomposition: any SPD matrix $A \in \mathbb{R}^{d\times d}$ can be written as $A = LL^\top$, where L is lower-triangular with positive entries along its diagonal. Reversely, for any lower-triangular L with positive diagonal, $LL^\top$ is SPD. Thus, to represent an SPD $A \in \mathbb{R}^{d\times d}$, we define $A(L) := LL^\top$ and parameterize $L(\theta)$ to be lower-triangular, have positive diagonal, and be differentiable in the parameters θ:

$$(L(\theta))_{ij} := \begin{cases} 0 & \text{if } i < j, \\ e^{\theta_{d(d-1)/2+i}} & \text{if } i = j, \\ \theta_{(i-2)(i-1)/2+j} & \text{if } i > j, \end{cases} \tag{4}$$

where $\theta \in \mathbb{R}^{d(d+1)/2}$.

Algorithm 2: Optimized Kalman Filter (OKF)

```
Input: training data $\{(x_{k,t}, z_{k,t})\}_{k=1}^{K}$ (Definition 1);
batch size $b$; loss function (e.g., MSE);
optimization_step function (e.g., Adam)
$d_x \leftarrow \text{len}(x_{1,1})$,   $d_z \leftarrow \text{len}(z_{1,1})$
Initialize $\theta_Q \in \mathbb{R}^{\frac{1}{2}d_x(d_x+1)}$, $\theta_R \in \mathbb{R}^{\frac{1}{2}d_z(d_z+1)}$
while training not finished do
    // Get $Q, R$ using Eq. (4)
    $\hat{Q} \leftarrow L(\theta_Q)L(\theta_Q)^\top$,   $\hat{R} \leftarrow L(\theta_R)L(\theta_R)^\top$
    $\mathcal{K} \leftarrow \text{sample}(\{1, ..., K\}, \text{size}=b)$
    $C \leftarrow 0$
    for $k$ in $\mathcal{K}$ do
        Initialize $\hat{x} \in \mathbb{R}^{d_x}$
        for $t$ in $1 : T_k$ do
            // KF steps (Fig. 1)
            $\hat{x} \leftarrow \text{KF\_predict}(\hat{x};\, \hat{Q})$
            $\hat{x} \leftarrow \text{KF\_update}(\hat{x}, z_{k,t};\, \hat{R})$
            $C \leftarrow C + \text{loss}(\hat{x},\, x_{k,t})$
    $\theta_Q, \theta_R \leftarrow \text{optimization\_step}(C,\, (\theta_Q, \theta_R))$
Return $\hat{Q}$, $\hat{R}$
```

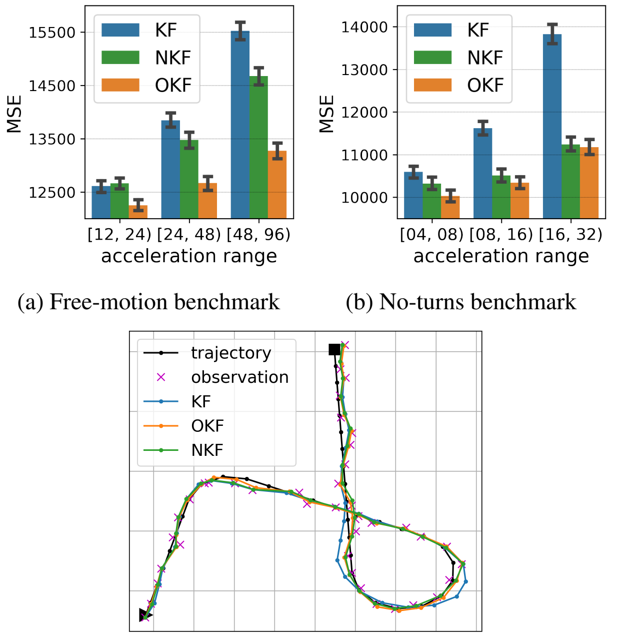
Both Cholesky parameterization and sequential optimization methods are well known tools. Yet, for KF optimization *from supervised data*, we are not aware of any previous attempts to apply them together, as noise estimation (Algorithm 1) is typically preferred. We wrap the optimization process in the Optimized KF (**OKF**) in Algorithm 2, which outputs optimized parameters $\hat{Q}, \hat{R}$ for Fig. 1.

Note that global optimality is guaranteed by neither noise estimation (Algorithm 1) nor OKF (Algorithm 2) – if Assumption 1 cannot be trusted. However, OKF at least addresses the desired objective – while noise estimation suffers from goal-misalignment, as analyzed in Appendix B.

4. Neural KF: Is Non-Linearity Helpful?

In this section, we demonstrate that comparing an optimized neural network to a non-optimized baseline may lead to incorrect conclusions: the neural network may seem superior to the baseline, even if the complicated architecture has no added value to the problem.

The demonstration relies on our experiments in the Doppler radar problem, where the location of various targets is estimated from noisy Doppler radar measurements, as described



(a) Free-motion benchmark

(b) No-turns benchmark

(c) Sample trajectory

Figure 2: (a-b) Test errors and 95% confidence intervals, over targets with different accelerations. The middle acceleration range coincides with the training accelerations (24-48 in (a) and 8-16 in (b)), and the other ranges correspond to out-of-distribution generalization. (c) A sample trajectory (projected onto XY plane). The standard KF provides inaccurate predictions in certain turns.

in detail in Appendix B.1. For this problem, we developed a Neural Kalman Filter model (NKF), which incorporates an LSTM model into the KF framework. NKF was designed to improve sequential prediction under non-linear dynamics of highly-maneuvering targets, and we made honest efforts to engineer a well-motivated architecture for the problem, as presented in Appendix E and Fig. 13. Regardless, we stress that this section demonstrates a methodological flaw when comparing *any* optimized filtering method to the KF; this methodological argument stands regardless of the technical quality of NKF. In addition, Appendix E presents similar results for other variants of NKF.

Experiments: We train NKF and OKF on a dataset of simulated trajectories, representing realistic targets with free motion (as displayed in Fig. 2c). As a second benchmark, we also train on a dataset of simplified trajectories, with speed changes but with no turns. The two benchmarks are specified in detail in Appendix C.1, and correspond to Fig. 9d and Fig. 9e. We tune the KF from the same datasets using Algorithm 1. In addition to out-of-sample test trajectories, we also test generalization to out-of-distribution trajectories, generated using different ranges of target accelerations (affecting both speed changes and turns radiuses).

Fig. 2 summarizes the test results. Compared to KF, NKF reduces the errors in both benchmarks, suggesting that the non-linear architecture pays off. However, optimizing the KF (using OKF) reduces the errors even further, and thus reverses the conclusion. That is, the advantage of NKF in this problem comes *exclusively* from optimization, and *not at all* from the expressive architecture. Of course, this experiment does not evaluate neural networks in general; yet, in this example, without optimizing the KF, the over-complicated NKF architecture would be preferred unjustifiably.

5. OKF vs. KF

Section 4 presents the methodological contribution of OKF for non-linear filtering, as an optimized baseline for comparison, instead of the standard KF. In Appendices B,C, we study the advantage of OKF over the KF more generally. We demonstrate that **OKF consistently outperforms the KF** – in 3 different domains (radar, video and lidar), over different problem variations, using different KF baselines, with different data sizes, and even under distributional shifts.

This result carries considerable practical significance. In real-world deployed systems, moving from the KF to non-linear models requires deployment of a new architecture, often with additional overhead and risks, increased latency and increased complexity. However, **moving from the KF to OKF merely requires changing the parameters $\hat{Q}$, $\hat{R}$.**

Recall that by Theorem 1, the KF is already optimal when $\hat{Q} = Q$, $\hat{R} = R$, which seems to contradict the advantage of OKF. This contradiction is explained by the reliance of Theorem 1 on Assumption 1. In Appendix B, we extensively discuss the fragility of Assumption 1 and its violations. We show that certain violations may be arguably difficult to even *notice*, yet cause major changes in the optimal values of $\hat{Q}$ and $\hat{R}$ – both theoretically and empirically. This motivates the explicit optimization of the KF parameters using OKF – whenever Assumption 1 is in doubt.

6. Summary

In non-linear filtering, it is common to evaluate an optimized model against a non-optimized KF baseline. We demonstrated that this may produce misleading conclusions, and introduced the Optimized KF (OKF) as a fair baseline for comparison. OKF can also be used as a linear filtering algorithm instead of the standard KF, and was shown to consistently improve the accuracy in a variety of scenarios. OKF was further motivated by theoretical analysis of the KF sub-optimality. From a practical point of view, OKF is easily applicable to new problems. Since its architecture is identical to the KF, the learned model causes neither inference-time delays nor deployment overhead. All these properties make OKF a powerful practical tool for both linear and non-linear filtering problems.

References

Pieter Abbeel, Adam Coates, Michael Montemerlo, Andrew Ng, and Sebastian Thrun. Discriminative training of kalman filters. Robotics: Science and systems, pages 289–296, 06 2005.

S. Akhlaghi, N. Zhou, and Z. Huang. Adaptive adjustment of noise covariance in kalman filter for dynamic state estimation. In 2017 IEEE Power Energy Society General Meeting, pages 1–5, 2017.

Zafer Aydogmus and Omur Aydogmus. A comparison of artificial neural network and extended kalman filter based sensorless speed estimation. Measurement, 63:152–158, 2015. ISSN 0263-2241.

Yu-ting Bai, Xiao-yi Wang, Xue-bo Jin, Zhi-yao Zhao, and Bai-hai Zhang. A neuron-based kalman filter with nonlinear autoregressive model. Sensors, 20(1):299, 2020.

S. T. Barratt and S. P. Boyd. Fitting a kalman smoother to data. In 2020 American Control Conference (ACC), pages 1526–1531, 2020.

David K Barton. Modern radar system analysis. Norwood, 1988.

Stav Belogolovsky, Ido Greenberg, Danny Eitan, and Shie Mannor. Continuous forecasting via neural eigen decomposition of stochastic dynamics. arXiv preprint arXiv:2202.00117, 2022.

B. Carew and P. Belanger. Identification of optimum filter steady-state gain for systems with unknown noise covariances. IEEE Transactions on Automatic Control, 18(6): 582–587, 1973.

Huseyin Coskun, Felix Achilles, Robert DiPietro, Nassir Navab, and Federico Tombari. Long short-term memory kalman filters: Recurrent neural estimators for pose regularization. ICCV, 2017.

Patrick Dendorfer, Hamid Rezatofighi, Anton Milan, Javen Shi, Daniel Cremers, Ian Reid, Stefan Roth, Konrad Schindler, and Laura Leal-Taixé. Mot20: A benchmark for multi object tracking in crowded scenes, 2020. URL `https://motchallenge.net/data/MOT20/`.

Lichuan Deng, Da Li, and Ruifang Li. Improved IMM Algorithm based on RNNs. Journal of Physics Conference Series, 1518:012055, April 2020.

Jimmy Ba Diederik P. Kingma. Adam: A method for stochastic optimization, 2014. URL `https://arxiv.org/abs/1412.6980`.

Hai fa Dai, Hong wei Bian, Rong ying Wang, and Heng Ma. An ins/gnss integrated navigation in gnss denied environment using recurrent neural network. Defence Technology, 16(2):334–340, 2020. ISSN 2214-9147.

B. Feng, M. Fu, H. Ma, Y. Xia, and B. Wang. Kalman filter with recursive covariance estimation—sequentially estimating process noise covariance. IEEE Transactions on Industrial Electronics, 61(11):6253–6263, 2014.

Simone Formentin and Sergio Bittanti. An insight into noise covariance estimation for kalman filter design. IFAC Proceedings Volumes, 47(3):2358–2363, 2014. ISSN 1474-6670. 19th IFAC World Congress.

Chang Gao, Junkun Yan, Shenghua Zhou, Bo Chen, and Hongwei Liu. Long short-term memory-based recurrent neural networks for nonlinear target tracking. Signal Processing, 164, 05 2019.

Matthieu Geist and Olivier Pietquin. Kalman filtering colored noises: the (autoregressive) moving-average case. In MLASA 2011, pages 1–4, Honolulu, United States, December 2011.

Sepp Hochreiter and Jurgen Schmidhuber. Long short-term memory. Neural Computation, 1997.

Jeffrey Humpherys, Preston Redd, and Jeremy West. A fresh look at the kalman filter. SIAM Review, 54(4): 801–823, 2012.

Ala A. Hussein. Kalman filters versus neural networks in battery state-of-charge estimation: A comparative study. International Journal of Modern Nonlinear Theory and Application, 2014.

Dan Iter, Jonathan Kuck, and Philip Zhuang. Target tracking with kalman filtering, knn and lstms, 2016.

Faisal Jamil et al. Toward accurate position estimation using learning to prediction algorithm in indoor navigation. Sensors, 2020.

Andrew H Jazwinski. Stochastic processes and filtering theory. Courier Corporation, 2007.

R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. Journal of Basic Engineering, 82 (1):35–45, 03 1960. ISSN 0021-9223.

Chanho Kim, Fuxin Li, and James M. Rehg. Multi-object tracking with neural gating using bilinear lstm. ECCV, September 2018.

Yaakov Bar-Shalom X.-Rong Li Thiagalingam Kirubarajan. Estimation with Applications to Tracking and Navigation: Theory, Algorithms and Software. John Wiley and Sons, Inc., 2002.

Tony Lacey. Tutorial: The kalman filter, 1998. URL "http://web.mit.edu/kirtley/kirtley/binlustuff/literature/control/Kalmanfilter.pdf".

S. Li, C. De Wagter, and G. C. H. E. de Croon. Unsupervised tuning of filter parameters without ground-truth applied to aerial robots. IEEE Robotics and Automation Letters, 4(4):4102–4107, 2019.

Huajun Liu, Hui Zhang, and Christoph Mertz. Deepda: Lstm-based deep data association network for multi-targets tracking in clutter. CoRR, abs/1907.09915, 2019a.

Jingxian Liu, Zulin Wang, and Mai Xu. Deepmtt: A deep learning maneuvering target-tracking algorithm based on bidirectional lstm network. Information Fusion, 53, 06 2019b.

R. Mehra. On the identification of variances and adaptive kalman filtering. IEEE Transactions on Automatic Control, 15(2):175–184, 1970.

A. Paulo Moreira, Paulo Costa, and José Lima. New approach for beacons based mobile robot localization using kalman filters. Procedia Manufacturing, 51:512–519, 2020. 30th International Conference on Flexible Automation and Intelligent Manufacturing (FAIM2021).

Dominic A. Neu, Johannes Lahann, and Peter Fettke. A systematic literature review on state-of-the-art deep learning methods for process prediction. CoRR, abs/2101.09320, 2021.

Brian Odelson, Alexander Lutz, and James Rawlings. The autocovariance least-squares method for estimating covariances: Application to model-based control of chemical reactors. Control Systems Technology, IEEE Transactions on, 14:532 – 540, 06 2006.

Sebin Park et al. Measurement noise recommendation for efficient kalman filtering over a large amount of sensor data. Sensors, 2019.

José C. Pinheiro and Douglas M. Bates. Unconstrained parameterizations for variance-covariance matrices. Statistics and Computing, 6:289–296, 1996.

Guy Revach, Nir Shlezinger, Xiaoyong Ni, Adria Lopez Escoriza, Ruud JG Van Sloun, and Yonina C Eldar. Kalmannet: Neural network aided kalman filtering for partially known dynamics. IEEE Transactions on Signal Processing, 70:1532–1547, 2022.

Anirban Roy and Debjani Mitra. Multi-target trackers using cubature kalman filter for doppler radar tracking in clutter. IET Signal Processing, 10(8):888–901, 2016.

David E. Rumelhart et al. Learning representations by back-propagating errors. Nature, 1986.

A. Sengupta, F. Jin, and S. Cao. A dnn-lstm based target tracking approach using mmwave radar and camera sensor fusion. 2019 IEEE National Aerospace and Electronics Conference (NAECON), pages 688–693, 2019.

Robert H. Shumway and David S. Stoffer. Time Series Analysis and Its Applications (Springer Texts in Statistics). Springer-Verlag, Berlin, Heidelberg, 2005. ISBN 0387989501.

Harold Wayne Sorenson. Kalman Filtering: Theory and Application. IEEE Press, 1985.

Israr Ullah, Muhammad Fayaz, and DoHyeun Kim. Improving accuracy of the kalman filter algorithm in dynamic conditions using ann-based learning module. Symmetry, 11(1), 2019. ISSN 2073-8994.

Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In 2017 IEEE International Conference on Image Processing (ICIP), pages 3645–3649, 2017.

L. Zanni, J. Le Boudec, R. Cherkaoui, and M. Paolone. A prediction-error covariance estimator for adaptive kalman filtering in step-varying processes: Application to power-system state estimation. IEEE Transactions on Control Systems Technology, 25(5):1683–1697, 2017.

Paul Zarchan and Howard Musoff. Fundamentals of Kalman Filtering: A Practical Approach. American Institute of Aeronautics and Astronautics, 2000.

Tianyu Zheng, Yu Yao, Fenghua He, and Xinran Zhang. An rnn-based learnable extended kalman filter design and application. In 2019 18th European Control Conference (ECC), pages 3304–3309, 2019.

Contents

A. Related Work

Noise estimation: Estimation of the KF noise parameters from observations alone has been studied for decades, as supervised data (Definition 1) is often unavailable. Various methods were studied, based on autocorrelation (Mehra, 1970; Carew and Belanger, 1973), EM (Shumway and Stoffer, 2005) and others (Odelson et al., 2006; Feng et al., 2014; Park et al., 2019). When supervised data *is* available, noise estimation reduces to Eq. (2) and is considered a solved problem (Odelson et al., 2006). We show that while noise estimation is indeed easy from supervised data, it is often not the right objective to pursue.

Many works addressed the problem of non-stationary noise estimation (Zanni et al., 2017; Akhlaghi et al., 2017). However, as demonstrated in Section 4, stationary methods may be highly competitive if tuned correctly – even in problems with complicated dynamics.

Optimization: We apply gradient-based optimization to the KF with respect to its errors. In absence of supervised data, gradient-based optimization was suggested for other losses, such as smoothness (Barratt and Boyd, 2020). In the supervised setting, noise estimation is typically preferred (Odelson et al., 2006), although optimization without gradients was suggested in Abbeel et al. (2005). In practice, "optimization" of KF is sometimes handled by trial and error (Jamil et al., 2020) or grid search (Formentin and Bittanti, 2014; Coskun et al., 2017). In other cases, Q and R are restricted to be diagonal (Li et al., 2019; Formentin and Bittanti, 2014). However, such heuristics may not suffice when the optimal parameters take a non-trivial form (such as their form in Proposition 1).

Neural Networks (NNs) in filtering: The NKF in Section 4 relies on a recurrent NN. NNs are widely used in non-linear filtering, e.g., for online prediction (Gao et al., 2019; Iter et al., 2016; Coskun et al., 2017; fa Dai et al., 2020; Belogolovsky et al., 2022), near-online prediction (Kim et al., 2018), and offline prediction (Liu et al., 2019b). Learning visual features for tracking via a NN was suggested by Wojke et al. (2017). NNs were also considered for related problems such as data association (Liu et al., 2019a), model switching (Deng et al., 2020), and sensors fusion (Sengupta et al., 2019).

In addition, all the 10 studies cited in Section 1 used a NN model for non-linear filtering, with either KF or EKF as a baseline for comparison. As discussed above, none has optimized the baseline model to a similar extent as the NN. As demonstrated in Section 4, such experimental methodology could lead to unjustified conclusions.

B. OKF vs. KF

Section 4 presents the methodological contribution of OKF for non-linear filtering, as an optimized baseline for comparison, instead of the standard KF. In this section, we study the advantage of OKF over the KF more generally. We demonstrate that OKF consistently outperforms the KF in a variety of scenarios from 3 different domains. This result carries high practical significance, since shifting from KF to OKF in real-world deployed systems only requires change of the parameters $\hat{Q},\ \hat{R}$.

Recall that by Theorem 1, the KF may provide inferior accuracy only if Assumption 1 is violated. Thus, the violations are discussed in depth, and the effects of certain violations are analyzed theoretically.

B.1. Doppler Radar Tracking

The Doppler radar problem: We consider a variant of the classic Doppler radar problem (Barton, 1988; Roy and Mitra, 2016), where various targets trajectories are tracked in a homogeneous 3D space, given regular observations of a Doppler radar. The state $X = (x_x, x_y, x_z, x_{ux}, x_{uy}, x_{uz})^\top \in \mathbb{R}^6$ consists of 3D location and velocity. The goal is to minimize the MSE over the 3 location coordinates. While the true dynamics F are unknown to the KF, a constant-velocity model $\tilde{F}$ can be used:

$$\tilde{F} = \begin{pmatrix} 1 & & & 1 & & \\ & 1 & & & 1 & \\ & & 1 & & & 1 \\ & & & 1 & & \\ & & & & 1 & \\ & & & & & 1 \end{pmatrix}. \tag{5}$$

An observation $Z \in \mathbb{R}^4$ consists of the location in spherical coordinates (range, azimuth, elevation) and the radial velocity (the Doppler signal), with an additive i.i.d Gaussian noise. After transformation to Cartesian coordinates, the observation model can be written as:

$$H = H(X) = \begin{pmatrix} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & \frac{x_x}{r} & \frac{x_y}{r} & \frac{x_z}{r} \end{pmatrix}, \tag{6}$$

where $r = \sqrt{x_x^2 + x_y^2 + x_z^2}$. Since $H = H(X)$ relies on the unknown location (x_x, x_y, x_z), we instead substitute $\tilde{H} := H(Z)$ in the KF update step in Fig. 1.

Assumption violations in the Doppler problem: Theorem 1 guarantees the optimality of the KF parameters estimated by Algorithm 1. Yet, in Section 4, OKF outperforms the KF. This result is made possible by the violation of Assumption 1: while the Doppler radar problem of Section 4 may not seem complex, the trajectories follow a non-linear motion model (as displayed in Fig. 2c).

Imagine that we simplified the problem from Section 4 by only simulating constant-velocity targets, making the true motion model F linear. Would this recover Assumption 1 and make OKF unnecessary? The answer is *no*; the adventurous reader may attempt to list all the remaining violations before reading on.

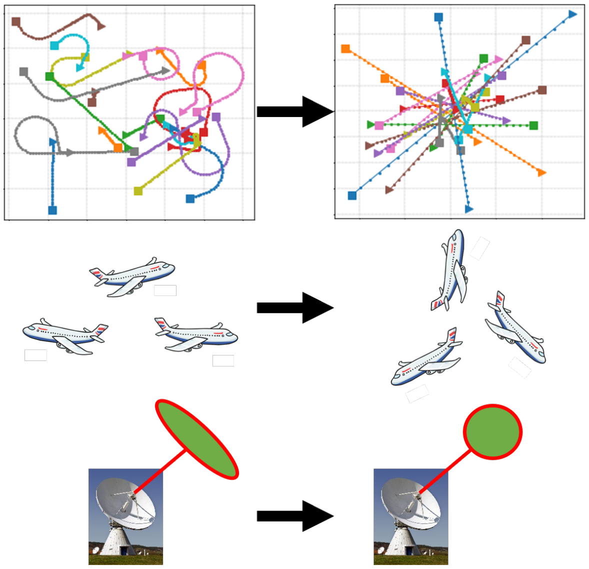


Figure 3: The toy benchmark is simplified with linear motion, isotropic flying directions and physically-impossible radar. After all the simplifications, Assumption 1 still does not hold, thus Algorithm 1 is still sub-optimal and outperformed by OKF.

The simulated targets move mostly horizontally, with limited elevation changes. This is not expressed by the KF's initial state distribution $(\hat{x}_0, \tilde{P}_0)$. To remedy this, one may simulate motion uniformly in all directions. A third violation comes from the observation noise. While the radar noise is i.i.d in spherical coordinates (as mentioned in Section 4), it is not i.i.d in *Cartesian* coordinates (see discussion in Appendix D.2). To overcome this, one may simulate a radar with (physically-impossible) Cartesian i.i.d noise. This results in the unrealistically-simplified problem visualized in Fig. 3.

Despite the simplifications, it turns out that Assumption 1 is still not met, as the observation model in Eq. (6) is still not linear (i.e., $H = H(X)$ is not constant). As shown by Proposition 1, this single violation alone results in a significant deviation of Algorithm 1 from the optimal parameters.

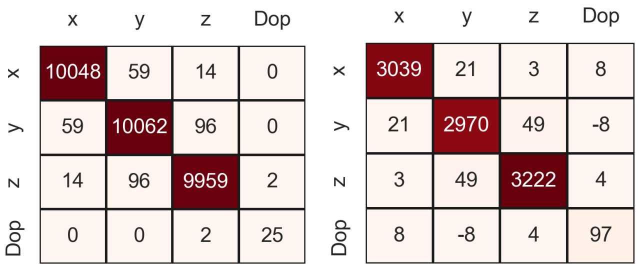
We first define the simplified problem.

Problem 1 (The toy Doppler problem)**.** The toy Doppler problem is the filtering problem modeled by Eq. (1), with constant-velocity dynamics F (Eq. (5)), Doppler observation H (Eq. (6)), and

$$Q = \mathbf{0} \in \mathbb{R}^{6\times 6}, \qquad R = \begin{pmatrix} \sigma_x^2 & & & \\ & \sigma_y^2 & & \\ & & \sigma_z^2 & \\ & & & \sigma_D^2 \end{pmatrix},$$

where $\sigma_x, \sigma_y, \sigma_z, \sigma_D > 0$.

Recall that $H = H(X)$ in Eq. (6) depends on the state X, which is unknown to the model. Thus, we assume that $\tilde{H} = H(\tilde{X})$ is used in the KF update step (Fig. 1), with some estimator $\tilde{X} \approx X$ (e.g., $\tilde{H} = H(Z)$ in Section 4). Hence, the *effective*



	x	y	z	Dop
x	10048	59	14	0
y	59	10062	96	0
z	14	96	9959	2
Dop	0	0	2	25

(a) KF / R

	x	y	z	Dop
x	3039	21	3	8
y	21	2970	49	-8
z	3	49	3222	4
Dop	8	-8	4	97

(b) OKF / R

Figure 4: The parameters $\hat{R}$ learned by KF and OKF in the toy Doppler problem. In each matrix axis, the entries correspond to the location (x, y, z) and the radial velocity ($Doppler$). The simulated noise variance is 100^2 for the positional dimensions and 5^2 for velocity, and is estimated accurately by the KF. However, OKF increases the noise associated with velocity, in accordance with Proposition 1. The decrease in the positional variance comes from scale-invariance in the toy problem, as discussed in Appendix D.1.

noise is $\tilde{R} := Cov(Z - \tilde{H}X) \neq Cov(Z - HX) = R$. Proposition 1 analyzes the difference between $\tilde{R}$ and R. To simplify the analysis, we further assume that the error $\tilde{X} - X$ within $\tilde{H}$ (e.g., $Z - X$) is independent of the target velocity.

Proposition 1. In the toy Doppler Problem 1 with the estimated observation model $\tilde{H}$, the effective observation noise $\tilde{R} = Cov(Z - \tilde{H}X)$ is:

$$\tilde{R} = \begin{pmatrix} \sigma_x^2 & & & \\ & \sigma_y^2 & & \\ & & \sigma_z^2 & \\ & & & \sigma_D^2 + C \end{pmatrix} = R + \begin{pmatrix} 0 & & & \\ & 0 & & \\ & & 0 & \\ & & & C \end{pmatrix}, \tag{7}$$

where $C = \Omega(\mathbb{E}[||u||^2])$ is the asymptotic lower bound ("big omega") of the expected square velocity u. In particular, $C > 0$ and is unbounded as the typical velocity grows.

Proof sketch (see complete proof in Appendix D.1). We have $Cov(Z - \tilde{H}X) = Cov(Z - HX + (H - \tilde{H})X) = R + Cov((H - \tilde{H})X)$, where the last equality relies on the independence between the target velocity and the estimation error $\tilde{X} - X$. We then calculate $Cov((H - \tilde{H})X)$. □

Proposition 1 has an intuitive interpretation: when measuring the velocity, Algorithm 1 only considers the inherent Doppler signal noise σ_D. However, the *effective* noise $\sigma_D + C$ also includes the *transformation error* from Doppler to the Cartesian coordinates, caused by the uncertainty in $H(X)$ itself. Notice that heuristic solutions such as inflation of R would not recover the effective noise $\tilde{R}$, which only differs from R in one specific entry. Yet, as demonstrated below, while Algorithm 1 learns to use $\hat{R} \approx R$, OKF captures the effective noise $\tilde{R}$ successfully.

Experiments: We test KF and OKF on the toy Problem 1 using the same methodology as in Section 4. In accordance with Proposition 1, OKF adapts the Doppler noise parameter: as shown in Fig. 4, it increases σ_D in proportion to the location noise by a factor of ≈ 13. Note that we refer to the proportion instead of absolute values due to scale-invariance in the toy problem, as discussed in Appendix D.1. Following the optimization, **OKF reduces the test MSE from 152 to 84 – a reduction of 44%**.

In this toy problem, the optimal parameters could in fact be derived analytically from Proposition 1. In practical problems, however, analytical solution is often infeasible. In fact, as discussed above, even specifying the model itself is not always trivial. Clearly, analytical solution of the wrong model would result in unaware sub-optimality. Instead, OKF optimizes the prediction errors directly from data, without any prior knowledge of the model.

Extended experiments: This section and Section 4 test OKF against KF in three specific variants of the Doppler problem. One may wonder if OKF's advantage generalizes to other scenarios, such as:

- Different subsets of violations of Assumption 1;
- Other baseline models than KF, e.g., the Extended KF;
- Small training datasets;
- Generalization to out-of-distribution test data.

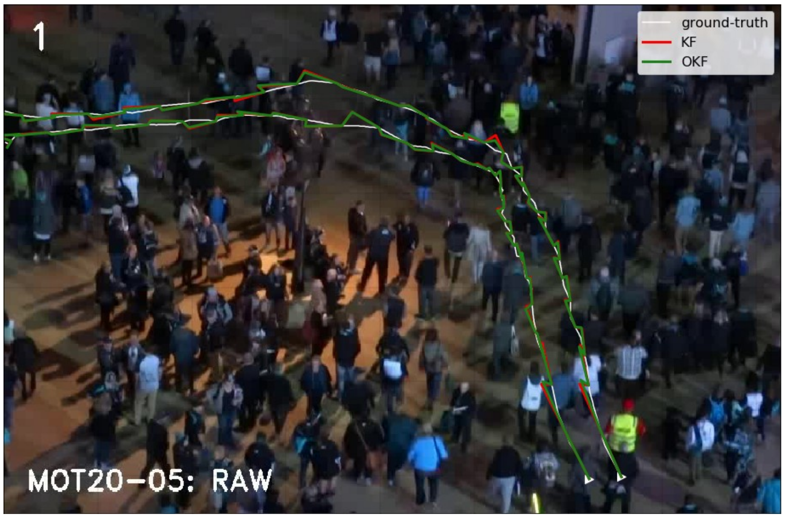
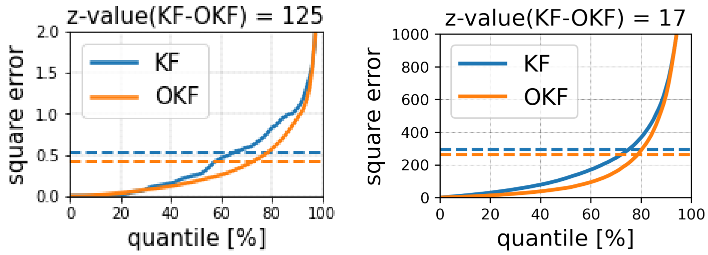


Figure 5: A sample of 2 trajectories in the first frame of MOT20 test video, along with the predictions of KF and OKF.



(a) Video tracking

(b) Lidar-based state estimation

Figure 6: Summary of test errors. The dashed lines correspond to MSE. Both z-values correspond to p-value $< 10^{-6}$. Each z-value is calculated over the N test trajectories as follows: $z = \frac{\mathrm{mean}(\{\Delta_i\})}{\mathrm{std}(\{\Delta_i\})}\sqrt{N}$, where $\Delta_i = err_i(KF)^2 - err_i(OKF)^2$ is the square-error difference on trajectory $1 \leq i \leq N$.

The extended experiments in Appendix C address *all* of the concerns above by examining a wide range of problem variations in the Doppler radar domain. In addition, other domains are experimented below. **In all of these experiments, OKF outperforms Algorithm 1 in terms of MSE**.

Finally, recall that if Assumption 1 is violated, Algorithm 1 is not aligned with the MSE objective. Interestingly, Appendix C.2 shows that this may cause Algorithm 1 to **deteriorate with the data size**.

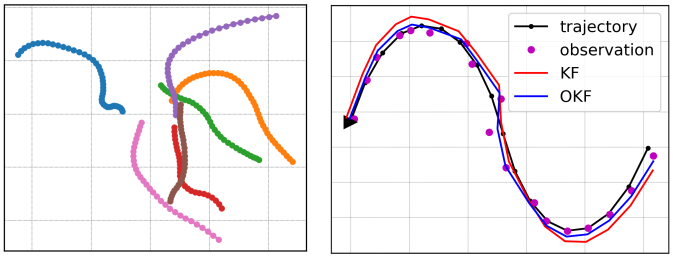
B.2. Video Tracking

The MOT20 dataset (Dendorfer et al., 2020) contains videos of real-world targets (mostly pedestrians, as shown in Fig. 5), along with their true location and size in every frame. For our experimental setup, since object detection is out of the scope, we assume that the true locations are known in real-time. The objective is to predict of the target location in the next frame. The state space corresponds to the 2D location, size and velocity, and the observations include only the location and size. The underlying dynamics F of the pedestrians are naturally unknown, and the standard constant-velocity model is used for $\tilde{F}$. This results in the following model:

$$\tilde{F} = \begin{pmatrix} 1 & & & & 1 & \\ & 1 & & & & 1 \\ & & 1 & & & \\ & & & 1 & & \\ & & & & 1 & \\ & & & & & 1 \end{pmatrix}, \quad \tilde{H} = H = \begin{pmatrix} 1 & & & & 0 & 0 \\ & 1 & & & 0 & 0 \\ & & 1 & & 0 & 0 \\ & & & 1 & 0 & 0 \end{pmatrix}.$$

Notice that the known observation model $\tilde{H} = H$ is *linear* (H is independent of X), hence poses a substantial difference from Appendix B.1 in terms of violations of Assumption 1.

The first three videos with 1117 trajectories are used for training, and the last video with 1208 trajectories for testing. As



(a) (b)

Figure 7: (a) A sample of simulated self-driving trajectories. (b) Segments of turns within a sample trajectory, and the corresponding lidar-based estimations.

shown in Fig. 6a, OKF reduces the test MSE by 18% with high statistical significance.

B.3. Lidar-based State Estimation in Self Driving

Consider the problem of state-estimation in self-driving, based on lidar measurements with respect to known landmarks (Moreira et al., 2020). The objective is to estimate the current vehicle location. We assume a single landmark (since the landmark matching problem is out of scope). We simulate driving trajectories consisting of multiple segments, with different accelerations and turn radiuses (Fig. 7a). The state is the vehicle's 2D location and velocity, and $\tilde{F}$ is modeled according to constant-velocity. The observation (both true H and modeled $\tilde{H}$) corresponds to the location, with an additive Gaussian i.i.d noise in polar coordinates. This results in the following model:

$$\tilde{F} = \begin{pmatrix} 1&0&1&0 \\ 0&1&0&1 \\ 0&0&1&0 \\ 0&0&0&1 \end{pmatrix}, \quad \tilde{H} = H = \left(\begin{smallmatrix} 1&0&0&0 \\ 0&1&0&0 \end{smallmatrix}\right).$$

We train KF and OKF over 1400 trajectories and test them on 600 trajectories. As shown in Fig. 6b, OKF reduces the test MSE by 10% with high statistical significance.

Notice that the lidar problem differs from Appendix B.1 in the linear observation model H, and from Appendix B.2 in the additive noise. Both properties have a major impact on the problem, as analyzed in Proposition 1 and below, respectively.

Theoretical analysis: As mentioned in Appendix B.1 and discussed in Appendix D.2, the i.i.d noise in polar coordinates is not i.i.d in Cartesian coordinates. In contrast to Appendix B.1, the observation model is linear this time. We seize the opportunity to isolate the i.i.d violation and study its effect. First, we define a simplified toy model – with simplified states, no-motion model F, isotropic motion noise Q and only radial observation noise.

Problem 2 (The toy lidar problem)**.** The toy lidar problem is the filtering problem modeled by Eq. (1) with the following parameters:

$$F = H = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \; Q = \begin{pmatrix} q & 0 \\ 0 & q \end{pmatrix}, \; R_{polar} = \begin{pmatrix} r_0 & 0 \\ 0 & 0 \end{pmatrix},$$

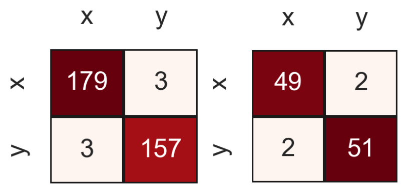
for some unknown $q, r_0 > 0$, with observation noise drawn i.i.d from $\mathcal{N}(0, R_{polar})$ in *polar* coordinates. The initial state X_0 follows a radial distribution (i.e., with a PDF of the form $f(||x_0||)$).

Proposition 2. As the number N of train trajectories in Problem 2 grows, the noise parameter $\hat{R}_N(KF)$ estimated by Algorithm 1 converges almost surely:

$$\hat{R}_N(KF) \xrightarrow{\text{a.s.}} \hat{R}_{est} = \begin{pmatrix} r_0/2 & 0 \\ 0 & r_0/2 \end{pmatrix}.$$

On the other hand, under regularity assumptions, the MSE is minimized by the parameter $\hat{R}_{opt} = \left(\begin{smallmatrix} r & 0 \\ 0 & r \end{smallmatrix}\right)$, where $r < r_0/2$.

Proof sketch (see complete proof in Appendix D.2). For $\hat{R}_{est}$, we calculate $\mathbb{E}[\hat{R}_N(KF)]$ and use the law of large numbers. For the calculation, we transform R_{polar} to Cartesian coordinates using the random direction variable θ, and take the expectation over $\theta \sim U([0, 2\pi))$. The uniform distribution of θ comes from the radial symmetry of the problem. For $\hat{R}_{opt}$, we calculate and minimize the expected square error directly. □



(a) KF / R (b) OKF / R

Figure 8: The parameters $\hat{R}$ learned by KF and OKF in the lidar problem.

Intuitively, Proposition 2 shows that the non-i.i.d nature of the noise in Cartesian coordinates reduces the *effective* noise. Note that the analysis only holds for the unrealistic toy Problem 2. The empirical setting in this section is less simplistic, and the generalization of Proposition 2 is not trivial. Fortunately, OKF optimizes the noise parameters directly from the data, and does not require such theoretical analysis. Fig. 8 shows that indeed, in accordance with the intuition of Proposition 2, OKF learns to reduce the values of $\hat{R}$ in comparison to KF. This results in reduced test errors as specified above.

C. OKF: Extended Experiments

C.1. Additional Scenarios and Baselines: A Case Study

In this section, we extend the experiments of Appendix B.1 with a detailed case study. The case study considers 5 types of tracking scenarios (*benchmarks*) and 4 variants of the KF (*baselines*) – 20 experiments in total. In each experiment, we compare the test MSE of OKF against the standard KF. The experiments in Section 4 and Appendix B.1 are 3 particular cases. For each benchmark, we simulate 1500 targets for training and 1000 targets for testing.

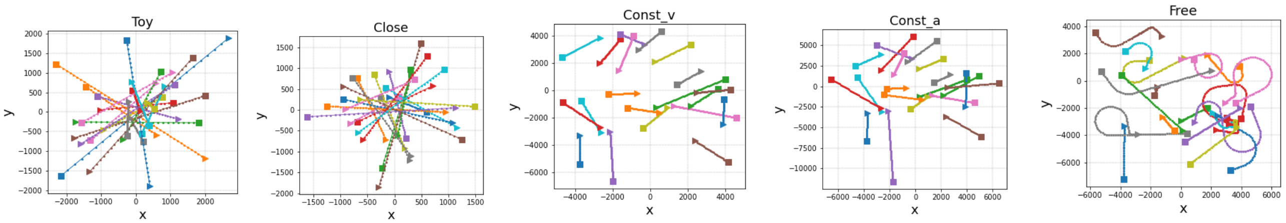
Benchmarks (scenarios): Appendix B discusses the sensitivity of Algorithm 1 to violations of Assumption 1. In this case study, we consider 5 benchmarks with different subsets of violations of Assumption 1. The *Free Motion* benchmark is intended to represent a realistic Doppler radar problem, with targets and observations simulated as in Section 4: each target trajectory consists of multiple segments of different turns and accelerations. On the other extreme, the *Toy* benchmark (Problem 1) introduces multiple simplifications (as visualized in Fig. 3). In the Toy benchmark, the only violation of Assumption 1 is the non-linear observation H, as discussed in Appendix B.1. Note that Appendix B.2 and Appendix B.3 experiment with settings of a linear observation model.

We design 5 benchmarks within the spectrum of complexity between Toy and Free Motion. Each benchmark is defined as a subset of the following properties, as specified in Table 1 and visualized in Fig. 9:

- *anisotropic*: horizontal motion is more likely than vertical (otherwise direction is distributed uniformly).
- *polar*: radar noise is generated i.i.d in spherical coordinates (otherwise noise is Cartesian i.i.d).
- *uncentered*: targets are dispersed in different locations far from the radar (otherwise they are concentrated in the center).
- *acceleration*: speed change is allowed (through intervals of constant acceleration).
- *turns*: non-straight motion is allowed.

Table 1: Benchmarks and the properties that define them. "V" means that the benchmark satisfies the property.

Benchmark	anisotropic	polar	uncentered	acceleration	turns
Toy	O	O	O	O	O
Close	V	V	O	O	O
Const_v	V	V	V	O	O
Const_a	V	V	V	V	O
Free	V	V	V	V	V



(a) Toy (b) Close (c) Const_v (d) Const_a (e) Free motion

Figure 9: Samples of targets trajectories in the various benchmarks, projected onto the XY plane.

Table 2: Test MSE results of Algorithm 1 and Algorithm 2 over 5 benchmarks (scenarios) and 4 baselines (variants of KF). For KFp we also consider an "oracle" baseline with perfect knowledge of the noise.

Benchmark	KF	OKF	KFp	KFp (oracle)	OKFp	EKF	OEKF	EKFp	OEKFp
Toy	151.7	84.2	269.6	–	116.4	92.8	**79.4**	123.0	109.1
Close	25.0	24.8	22.6	22.5	**22.5**	26.4	26.1	24.5	24.1
Const_v	90.2	90.0	102.3	102.3	**89.2**	102.5	99.7	112.7	102.1
Const_a	107.5	101.6	118.4	118.3	**100.3**	110.0	107.0	126.0	108.7
Free	125.9	118.8	145.6	139.3	**117.9**	135.8	121.9	149.3	120.0

Baselines (KF variants): All the experiments above compare OKF to the standard KF baseline. In practice, other variants of the KF are often in use. Here we define 4 such variants as different baselines to the experiments. In each experiment, we compare the baseline tuned by Algorithm 1 to its Optimized version trained by Algorithm 2 (denoted with the prefix "O" in its name). For Algorithm 2, we use the Adam optimizer with a single training epoch over the 1500 training trajectories, 10 trajectories per training batch, and learning rate of 0.01.

The different baselines are designed as follows. *EKF* baselines use the non-linear Extended KF model (Sorenson, 1985). The EKF replaces the approximation $H \approx H(z)$ of Section 4 with $H \approx \nabla_x h(\hat{x})$, where $h(x) = H(x) \cdot x$ and $\tilde{x}$ is the current state estimate. *Polar* baselines (denoted with "p") represent the observation noise R with spherical coordinates, in which the polar radar noise is i.i.d.

Results: Table 2 summarizes the test errors (MSE) in all the experiments. In each cell, the left column corresponds to the baseline Algorithm 1, and the right to Algorithm 2. In the model names, "O" stands for optimized, "E" for EKF and "p" for polar (or spherical). The same results are also shown with confidence intervals in Fig. 10. Below we discuss the main findings.

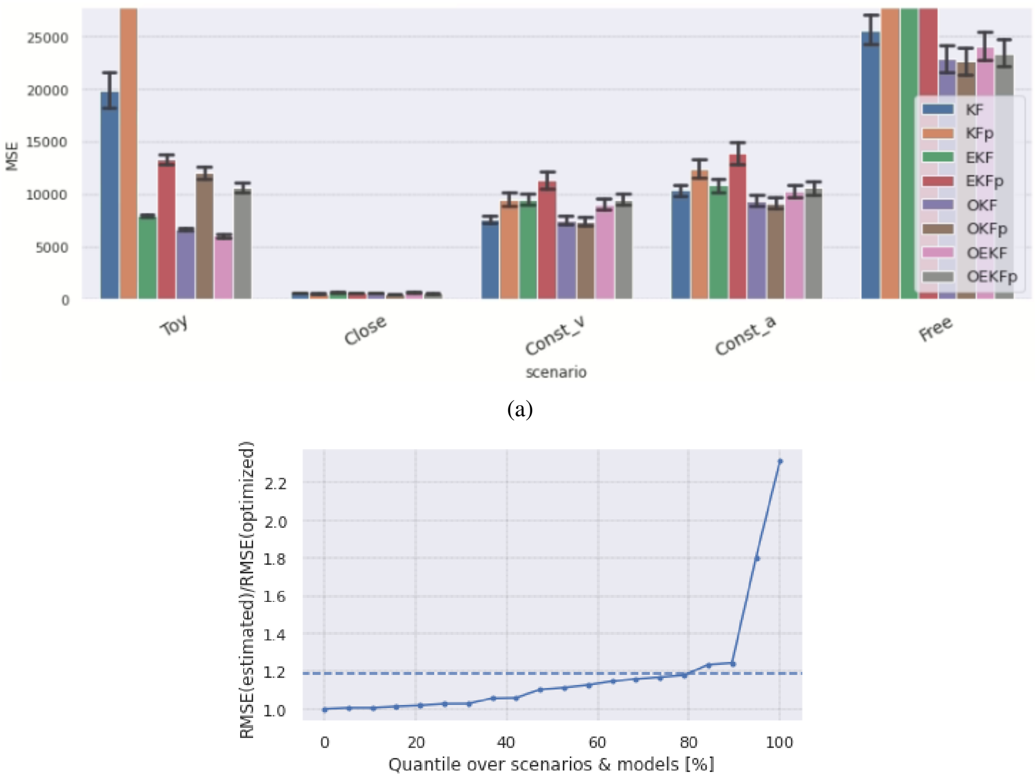
Choosing the KF configuration is not trivial: Consider the non-optimized KF baselines (left column in every cell in Table 2). In each benchmark, the results are sensitive to the baseline, i.e., to the choice of KF configuration – R's coordinates and whether to use EKF. For example, in the Toy benchmark, EKF is the best design, since the observation model H is non-linear. In other benchmarks, however, the winning baselines may come as a surprise:

1. Under non-isotropic motion direction (all benchmarks except Toy), EKF is worse than KF despite the non-linearity. It is possible that the horizontal prior reduces the stochasticity of H, making the derivative-based approximation unstable.
2. Even when the observation noise is spherical i.i.d, spherical representation of R is not beneficial when targets are scattered far from the radar (last 3 benchmarks). It is possible that with distant targets, Cartesian coordinates have a more important role in expressing the horizontal prior of the motion.

Since the best KF variant per benchmark seems hard to predict in advance, a practical system cannot rely on choosing the KF variant optimally – and should rather be robust to this choice.

OKF is more accurate *and* more baseline-robust: For *every* benchmark and *every* baseline (20 experiments in total), OKF (right column) outperformed noise estimation (left column). In addition, the variance between the baselines reduces under optimization, i.e., OKF makes the KF more robust to the selected configuration.

OKF outperform an *oracle* baseline: We designed an "oracle" KF baseline – with perfect knowledge of the observation noise covariance R in spherical coordinates. We used it for all benchmarks except for Toy (in which the radar noise is



(a)

(b)

Figure 10: Summary of the test MSE of Algorithm 1 and Algorithm 2 in different benchmarks (scenarios) and baselines. This is a different presentation of the results of Table 2. (a) also includes 95% confidence intervals. (b) shows, for each of the 20 experiments (5 benchmarks × 4 baselines), the MSE ratio between Algorithm 1 and Algorithm 2. We see that Algorithm 2 wins in *all* the experiments (ratio is always larger than 1) – in some cases by large margins. The dashed line represents the average MSE ratio over over all the experiments, showing an average advantage of 20% to OKF.

not generated in spherical coordinates). Note that in the constant-speed benchmarks (Close and Const_v), $Q = 0$ and is estimated quite accurately; hence, in these benchmarks the oracle has a practically perfect knowledge of both noise covariances. Nevertheless, the oracle yields very similar results to Algorithm 1. This indicates that **the benefit of OKF is not in a better estimation accuracy of Q and R, but rather in optimizing the desired objective**.

C.2. Sensitivity to Train Dataset Size

Each benchmark in the case-study of Appendix C.1 has 1500 targets in its train data. One may argue that numeric optimization may be more sensitive to smaller datasets than noise estimation; and even more so, when taking into account that the optimization procedure ”wastes” a portion of the train data as a validation set.

In this section we test this concern empirically, by repeating some of the experiments of Appendix C.1 with smaller subsets of the train datasets – beginning from as few as 20 training trajectories. Fig. 11 shows that the advantage of OKF over KF holds consistently for all sizes of train datasets, although it is indeed increases with the size. Interestingly, in the Free Motion benchmark, the test MSE of KF and KFp *increases* with the amount of train data!

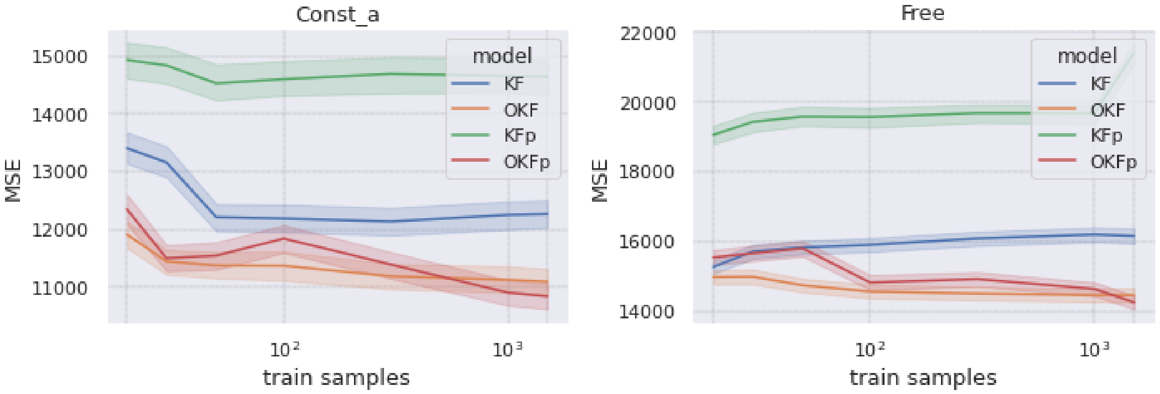


Figure 11: The advantage of OKF over KF holds consistently for all sizes of train datasets – including as small datasets as 20 trajectories. The shadowed areas correspond to 95% confidence intervals.

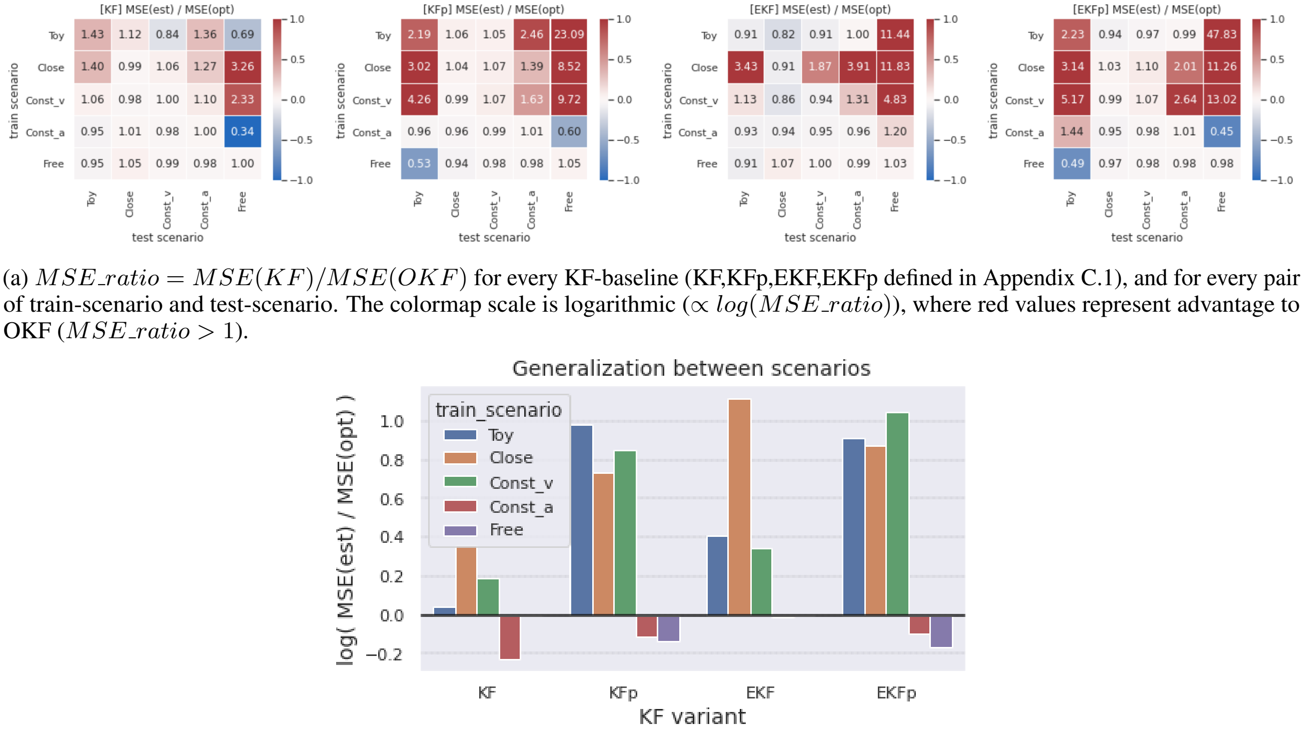
C.3. Generalization: Sensitivity to Distributional Shifts

In Appendix C.1, we demonstrate the robustness of OKF in different tracking scenarios: in every benchmark, OKF outperformed the standard KF over **out-of-sample test data**. This means that OKF did not overfit the noise in the training data. What about **out-of-distribution test data**? OKF learns patterns from the specific distribution of the train data – how well will it generalize to different distributions?

Section 4 already addresses this question to some extent, as OKF outperformes both KF and NFK over out-of-distribution target accelerations (affecting both speed changes and turns radius). In terms of Eq. (1), the modified acceleration corresponds to different magnitudes of motion noise Q; that is, we change the noise *after* OKF optimized the noise parameters. Yet, OKF adapted to the change without further optimization, with better accuracy than the standard KF. Thus, the results of Section 4 already provide a significant evidence for the robustness of OKF to certain distributional shifts.

In this section, we present a yet stronger evidence for the robustness of OKF – not over a parametric distributional shift, but over **entirely different benchmarks**. Specifically, we consider the 5 benchmarks (or scenarios) of Appendix C.1. For every pair (train-scenario, test-scenario), we train both KF and OKF on data of the train-scenario, then test them on data of the test-scenario. For every such pair of scenarios, we measure the generalization advantage of OKF over KF through $MSE\_ratio = MSE(KF)/MSE(OKF)$ (where $MSE\_ratio > 1$ indicates advantage to OKF). To measure the total generalization advantage of a model trained on a certain scenario, we calculate the geometric mean of MSE_ratio over all the test-scenarios (or equivalently, the standard mean over the logs of the ratios). The logarithmic scale guarantees a symmetric view of this metric of ratio between two scores.

This test is quite noisy, since a model optimized for a certain scenario may legitimately be inferior in other scenarios. Yet, considering all the results together in Fig. 12, it is evident that OKF provides more robust models: it generalizes better in most cases, sometimes by a large margin; and loses only in a few cases, always by a small margin.



(a) $MSE\_ratio = MSE(KF)/MSE(OKF)$ for every KF-baseline (KF,KFp,EKF,EKFp defined in Appendix C.1), and for every pair of train-scenario and test-scenario. The colormap scale is logarithmic ($\propto log(MSE\_ratio)$), where red values represent advantage to OKF ($MSE\_ratio > 1$).

(b) For every train-scenario, MSE_ratio is averaged over all the test-scenarios and is shown in a logarithmic scale. Positive values indicate advantage to OKF.

Figure 12: Generalization tests: OKF vs. KF under distributional shifts between scenarios.

D. Theoretical Analysis

D.1. Non-linear Observation

In this section, we discuss the relation between the theoretical analysis of Proposition 1 and the empirical results shown in Fig. 4. Then, we provide the proof of Proposition 1.

Fig. 4 vs. Proposition 1: Fig. 4 displays the noise parameters $\hat{R}$ learned by OKF in the toy problem. In accordance with Proposition 1, the noise σ_D associated with Doppler is increased compared to the true measurement noise R. In fact, not only σ_D is increased, but also the positional variances are decreased, which is not explained by Proposition 1. This phenomenon origins in the absence of dynamics noise in this toy problem ($Q \equiv 0$), which leads to scale-invariance w.r.t. the absolute values of $\hat{R}$. That is, if we multiply the whole matrix $\hat{R}$ by a constant factor, the filtering errors are unaffected. Specifically, if we multiply $\hat{R}$ of Fig. 4b by a factor of ≈ 3, the positional variances become aligned with those of Fig. 4a, and σ_D is increased by a factor of ≈ 13 – in accordance with Proposition 1. We repeated the tests with this modified $\hat{R}$, and indeed, the results were indistinguishable from the original OKF.

Proof of Proposition 1. Recall that in this problem, the KF applies the update step using an estimated observation model $\tilde{H} = H(\tilde{X})$:

$$\tilde{H} = \begin{pmatrix} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & \tilde{x}_x/\tilde{r} & \tilde{x}_y/\tilde{r} & \tilde{x}_z/\tilde{r} \end{pmatrix}.$$

Denoting the normalized estimation error $dx' = \frac{\tilde{x}}{\tilde{r}} - \frac{x}{r}$, we can rewrite $\tilde{H}$ as

$$\tilde{H} = H + \begin{pmatrix} 0 & & & & & \\ & 0 & & & & \\ & & 0 & & & \\ & & & dx'_x & dx'_y & dx'_z \end{pmatrix}.$$

By shifting the observation model in Eq. (1) from H to $\tilde{H}$, and denoting the noise by $\nu = (\nu_x, \nu_y, \nu_z, \nu_D)^\top$, we receive

$$Z = HX + \nu = \tilde{H}X + \begin{pmatrix} \nu_x \\ \nu_y \\ \nu_z \\ \nu_D - dx'_x u_x - dx'_y u_y - dx'_z u_z \end{pmatrix} = \tilde{H}X + \begin{pmatrix} \nu_x \\ \nu_y \\ \nu_z \\ \nu_D - dx' \cdot u \end{pmatrix},$$

where u denotes the current target velocity. We see that the effective observation noise is $\tilde{\nu} = Z - \tilde{H}X = (\nu_x, \nu_y, \nu_z, \nu_D - dx' \cdot u)^\top$.

To show that all the off-diagonal entries of $\tilde{R} = Cov(\tilde{\nu})$ vanish, recall that the estimation error dx' is assumed to be independent of the velocity u. According to Eq. (1), ν is also independent of u. Hence, $Cov(dx'_x \cdot u_x,\ \nu_x) = E(dx'_x \cdot u_x \cdot \nu_x) = E(dx'_x \nu_x) E(u_x)$ which vanishes by symmetry ($E(u_x) = 0$). The same result holds for coordinates y, z. Thus, $\tilde{R}$ is diagonal. Finally, by denoting $C = Var(dx' \cdot u) > 0$ we have $Cov(\tilde{\nu}) = \tilde{R}$ as required.

Relying again on symmetry $E(u), E(dx') = 0$, we can further calculate $C = Var(dx' \cdot u) = E(||dx'||^2)E(||u||^2) = \Omega(E(||u||^2))$, where Ω ("big-omega") corresponds to an asymptotic lower bound. □

D.2. Non-i.i.d Noise

The assumption of i.i.d noise in Assumption 1 is violated in many practical scenarios. Certain models with non-i.i.d noise can be solved analytically, if modeled correctly. For example, if the noise is auto-regressive with a known order p, an adjusted KF model may consider the last p values of the noise itself as part of the system state (Geist and Pietquin, 2011). However, the actual noise model is often unknown or infeasible to solve analytically.

Furthermore, the violation of the i.i.d assumption may even go unnoticed. We discuss a potential example in Appendix B, where the noise is i.i.d in *spherical* coordinates – but is not so after the transformation to *Cartesian* coordinates. To see that, consider a radar with noiseless angular estimation (i.e., only radial noise), and a low target ($x_z \approx 0$). Clearly, most of the noise concentrates on the XY plane – both in the current time-step and in the following ones (until the target moves away from the plane). Hence, the noise is statistically-dependent over time-steps.

We may formalize this intuition for the toy Problem 2. Denote the system state at time t by $X_t = ((X_t)_1, (X_t)_2)^\top$, and denote $\tan\theta_t = \frac{(X_t)_2}{(X_t)_1}$. By transforming R_{polar} of Problem 2 to Cartesian coordinates, the observation noise is drawn from the distribution $\nu_t \sim \mathcal{N}(0, R(\theta_t))$, where

$$R(\theta) = \begin{pmatrix} r_0\cos^2(\theta) & r_0\cos(\theta)\sin(\theta) \\ r_0\cos(\theta)\sin(\theta) & r_0\sin^2(\theta) \end{pmatrix}. \tag{8}$$

Since consecutive time steps are likely to have similar values of θ_t, the noise ν_t is no longer independent across time steps.

The effect of this violation of the i.i.d assumption is analyzed in Proposition 2, whose proof is provided below.

Proof of Proposition 2.

Noise estimation: First, notice that the whole setting of Problem 2 is invariant to the target direction θ: the initial state distribution is radial, and the motion noise Q is isotropic. Hence, for any target at any time-step, $\theta_t \sim [0, 2\pi)$ is uniformly distributed. By direct calculation,

$$\begin{aligned} E_\theta\left[\hat{R}_N(KF)_{11}\right] &= E_\theta\left[r_0\cos^2\theta\right] = \int_0^{2\pi} \frac{r_0}{2\pi}\cos^2\theta d\theta = \frac{r_0}{2} \\ E_\theta\left[\hat{R}_N(KF)_{22}\right] &= E_\theta\left[r_0\sin^2\theta\right] = \int_0^{2\pi} \frac{r_0}{2\pi}\sin^2\theta d\theta = \frac{r_0}{2} \\ E_\theta\left[\hat{R}_N(KF)_{12}\right] &= E_\theta\left[\hat{R}_N(KF)_{21}\right] = E_\theta\left[r_0\cos\theta\sin\theta\right] = 0. \end{aligned}$$

Since the targets in the data are i.i.d, the noise estimation of Algorithm 1 converges almost surely according to the law of large numbers, as required:

$$\hat{R}_N(KF) \xrightarrow{\text{a.s.}} \hat{R}_{est} = \begin{pmatrix} r_0/2 & 0 \\ 0 & r_0/2 \end{pmatrix}.$$

Optimization: We use again the radial symmetry and invariance to rotations in the problem: w.l.o.g, we assume that the optimal noise covariance parameter is diagonal, i.e., $\hat{R}_{opt}(r) = \left(\begin{smallmatrix} r & 0 \\ 0 & r \end{smallmatrix}\right)$ for some $r > 0$. Our goal is to find r, and in particular to compare it to $r_0/2$.

At a certain time t, where the system state is X_t, denote $E[X_t] = x_0 = (x_1, x_2)^\top$ and $Cov(X_t) = P_0 = \left(\begin{smallmatrix} p & 0 \\ 0 & p \end{smallmatrix}\right)$ (where $p > 0$). Denote the observation received at time t by $z = (x_1 + dx_1, x_2 + dx_2)^\top$. We are interested in the point-estimate $\hat{x}$ of the KF following the update step (Fig. 1). By substituting x_0, P_0, the observation z and the noise parameter $\hat{R}_{opt}(r)$ in the update step, we have

$$\begin{aligned} \hat{x} &= x_0 + P_0H^\top(HP_0H^\top + \hat{R}_{opt}(r))^{-1}(z - Hx_0) = x_0 + P_0(P_0 + \hat{R}_{opt}(r))^{-1}(z - x_0) \\ &= x_0 + \begin{pmatrix} \frac{p}{p+r} & 0 \\ 0 & \frac{p}{p+r} \end{pmatrix}\begin{pmatrix} dx_1 \\ dx_2 \end{pmatrix} = \begin{pmatrix} x_1 + \frac{p}{p+r}dx_1 \\ x_2 + \frac{p}{p+r}dx_2 \end{pmatrix}. \end{aligned}$$

On the other hand, the *true* observation noise covariance at time t is $R(\theta_t)$ of Eq. (8) (for the random variable θ_t). If we add the assumption that the state X_t is normally distributed ($X_t \sim \mathcal{N}(x_0, P_0)$), and use the true noise covariance $R(\theta_t)$, then the update step of Fig. 1 gives us the true posterior expected state:

$$\begin{aligned} x_{true} &= x_0 + P_0(P_0 + R(\theta_t))^{-1}(z - x_0) \\ &= x_0 + \begin{pmatrix} \frac{r_0\sin^2\theta + p}{p + r_0} & -\frac{r_0\cos\theta\sin\theta}{p + r_0} \\ -\frac{r_0\cos\theta\sin\theta}{p + r_0} & \frac{r_0\cos^2\theta + p}{p + r_0} \end{pmatrix}\begin{pmatrix} dx_1 \\ dx_2 \end{pmatrix} \\ &= \begin{pmatrix} x_1 + \frac{(r_0\sin^2\theta + p)dx_1 - (r_0\cos\theta\sin\theta)dx_2}{p + r_0} \\ x_2 + \frac{(r_0\cos^2\theta + p)dx_2 - (r_0\cos\theta\sin\theta)dx_1}{p + r_0} \end{pmatrix}. \end{aligned}$$

We can use the standard MSE decomposition for the point-estimate x, into the bias term of x and the variance term of the state distribution: $MSE = MSE_{var}(P_{true}) + MSE_{bias}(x, x_{true})$. Notice that $MSE_{var}(P_{true})$ is independent of our estimator, as it corresponds to the inherent uncertainty P_{true} (defined by applying to P_0 the update step with the true covariance $R(\theta_t)$). Thus, our objective is to minimize $MSE_{bias}(\hat{x}, x_{true}) = E[||\hat{x} - x_{true}||^2]$.

For the calculation below, we denote $a(r) := p/(p+r)$ and use the identity $\sin 2\theta = 2\cos\theta\sin\theta$. In addition, from radial symmetry of $dx = z - x_0$ we have $E[dx_1^2] = E[dx_2^2]$ and $E[dx_i] = 0$, thus we can denote $v := Var(dx_i) = E[dx_i^2]$.

$$\begin{aligned}
MSE_{bias}(\hat{x}(a), x_{true}) &= E||\hat{x}(a) - x_{true}||^2 \\
=&E\left[\left((a - \frac{r_0\sin^2\theta + p}{p+r_0})dx_1 + \frac{r_0\sin(2\theta)/2}{p+r_0}dx_2\right)^2\right. \\
&+\left.\left((a - \frac{r_0\cos^2\theta + p}{p+r_0})dx_2 + \frac{r_0\sin(2\theta)/2}{p+r_0}dx_1\right)^2\right] \\
=&E\left[dx_1^2\left(a^2 - 2a\frac{r_0\sin^2\theta + p}{p+r_0} + C_1\right) + \frac{r_0^2\sin^2(2\theta)/4}{(p+r_0)^2}dx_2^2 + A_1 dx_1 dx_2\right. \\
&+\left. dx_2^2\left(a^2 - 2a\frac{r_0\cos^2\theta + p}{p+r_0} + C_2\right) + \frac{r_0^2\sin^2(2\theta)/4}{(p+r_0)^2}dx_1^2 + A_2 dx_1 dx_2\right] \\
=&2va^2 - 2va\frac{r_0+2p}{p+r_0} + v(C_1+C_2) + v\frac{r_0^2\sin^2(2\theta)/2}{(p+r_0)^2},
\end{aligned}$$

where $C_{1,2}$ are independent of a, and $A_{1,2}$ are multiplied by $E[dx_1dx_2] = 0$ and vanish. To minimize we calculate

$$0 = \frac{\partial MSE_{bias}(\hat{x}(a), x_{true})}{\partial a} = 4v \cdot a - 2v\frac{2p+r_0}{p+r_0},$$

which gives us

$$a = \frac{p + r_0/2}{p+r_0}.$$

Notice that MSE_{bias} clearly diverges as $|a| \to \infty$, hence the only critical point necessarily corresponds to a minimum of the MSE. Hence, the optimal MSE is given when substituting the following r in $\hat{R}_{opt}$:

$$r = p/a - p = \frac{p^2 + pr_0 - (p^2 + pr_0/2)}{p + r_0/2} = \frac{pr_0}{2p+r_0}.$$

Finally, recall that $(\hat{R}_{est})_{ii} = r_0/2$ and compare to r directly:

$$(\hat{R}_{est})_{ii} - (\hat{R}_{opt})_{ii} = r_0/2 - r = \frac{r_0^2/2}{2p+r_0} > 0.$$

□

E. Neural KF: Extended Discussion and Experiments

Preliminaries – RNN and LSTM: *Recurrent neural networks* (RNN) (Rumelhart et al., 1986) are neural networks that are intended to be iteratively fed with sequential data samples, and that pass information (the *hidden state*) over iterations. Every iteration, the hidden state is fed to the next copy of the network as part of its input, along with the new data sample. *Long Short Term Memory* (LSTM) (Hochreiter and Schmidhuber, 1997) is an architecture of RNN that is particularly popular due to the linear flow of the hidden state over iterations, which allows to capture memory for relatively long term. The parameters of a RNN are usually optimized in a supervised manner with respect to a training dataset of input-output pairs.

Neural Kalman Filter: We introduce the Neural Kalman Filter (NKF), which incorporates an LSTM model into the KF framework. The framework provides a probabilistic representation (rather than point estimate) and a separation between the prediction and update steps. The LSTM is an architecture of recurrent neural networks, and is a key component in many SOTA algorithms for non-linear sequential prediction (Neu et al., 2021). We use it for the non-linear motion prediction.

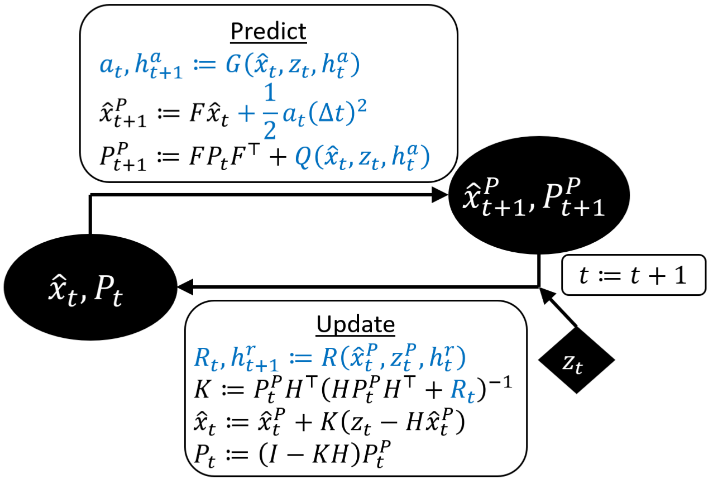

Predict
$a_t, h^a_{t+1} := G(\hat{x}_t, z_t, h^a_t)$
$\hat{x}^P_{t+1} := F\hat{x}_t + \frac{1}{2}a_t(\Delta t)^2$
$P^P_{t+1} := FP_tF^\top + Q(\hat{x}_t, z_t, h^a_t)$
$\hat{x}^P_{t+1}, P^P_{t+1}$
$t := t+1$
$\hat{x}_t, P_t$
z_t
Update
$R_t, h^r_{t+1} := R(\hat{x}^P_t, z^P_t, h^r_t)$
$K := P^P_t H^\top (HP^P_t H^\top + R_t)^{-1}$
$\hat{x}_t := \hat{x}^P_t + K(z_t - H\hat{x}^P_t)$
$P_t := (I - KH)P^P_t$

Figure 13: The Neural Kalman Filter (NKF). Differences from Fig. 1 are highlighted. Δt is constant; G, Q are the outputs of an LSTM network with hidden state h_a; and R is the output of an LSTM with hidden state h_r.

As shown in Fig. 13, NKF uses separate LSTM networks for prediction and update steps. In the prediction step, the target *acceleration* is predicted on top of the linear motion model, instead of predicting the state directly. This regularized formulation is intended to express our domain knowledge about the kinematic motion of physical targets.

Extended experiments: We extend the experiments of Section 4 with additional versions of NKF:

- Predicted-acceleration KF (**aKF**): a variant of NKF that predicts the acceleration but not the covariances Q and R.
- Neural KF (**NKF**): the model used in Section 4 and illustrated in Fig. 13.
- Neural KF with H-prediction (**NKFH**): a variant of NKF that also predicts the observation model H in every step.

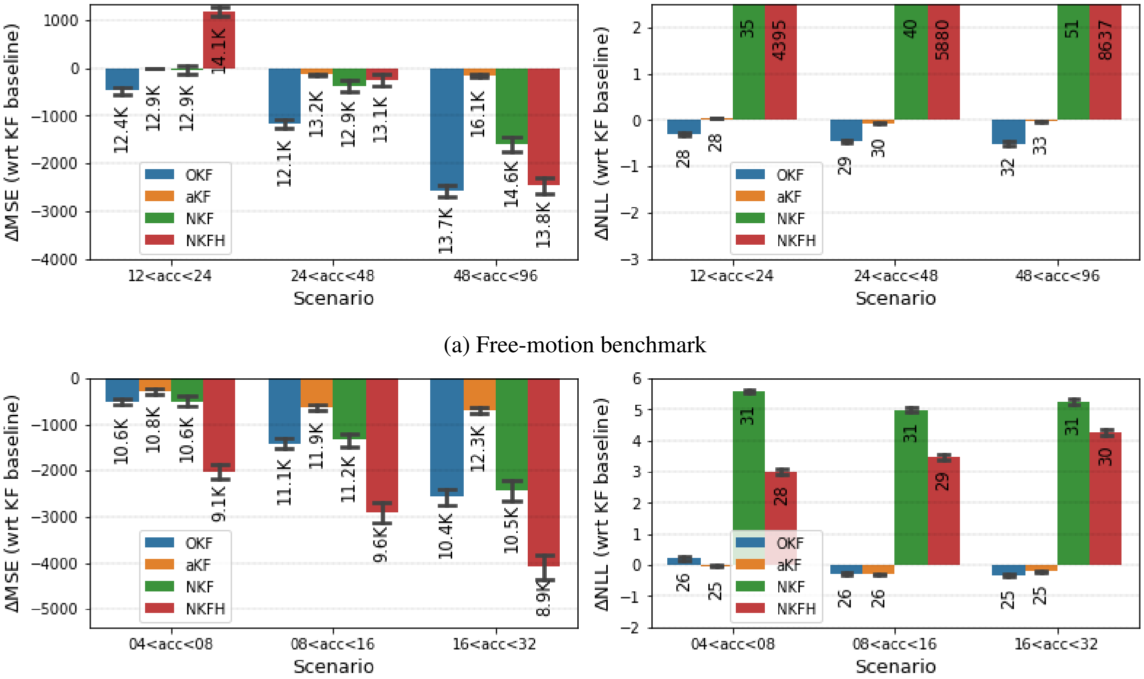
In addition, while we still train with MSE loss, we add the test metric of Negative-Log-Likelihood (NLL) – of the true state w.r.t the estimated distribution. Note that the NLL has an important role in the multi-target matching problem (which is out of the scope of this work).

For each benchmark and each model, we train the model on train data with a certain range of targets acceleration (note that acceleration affects both speed changes and turns sharpness), and tested it on targets with different acceleration ranges, some of them account for distributional shifts. For each model we train two variants – one with Cartesian representation of the observation noise R, and one with spherical representation (as in the baselines of Appendix C.1) – and we select the one with the higher validation MSE (where the validation data is a portion of the data assigned for training).

Fig. 14a shows that in the free-motion benchmark, all the 3 neural models improve the MSE in comparison to the standard KF, yet are outperformed by OKF. Furthermore, while OKF has the best NLL, the more complicated models NKF and NKFH increase the NLL in orders of magnitude. Note that the instability of NKFH is expressed in poor generalization to lower accelerations in addition to the extremely high NLL score.

Fig. 14b shows that in Const_a benchmark, all the 3 neural models improve the MSE in comparison to the standard KF, but only NKFH improves in comparison to OKF as well. On the other hand, NKFH still suffers from very high NLL.

In summary, all 3 variants of NKF outperform the standard KF in both benchmarks in terms of MSE. However, when comparing to OKF instead, aKF and NKF become inferior, and the comparison between NKFH and OKF depends on the selected benchmark and metric.



(a) Free-motion benchmark

(b) Const_a benchmark (no turns)

Figure 14: The *relative* MSE and NLL results of various models in comparison to the standard KF model. The textual labels specify the *absolute* MSE and NLL. Note that certain bars of NLL are of entirely different scale and thus are cropped in the figure (their values can be seen in the labels). In each benchmark, the models were trained with relation to MSE loss, on train data of the middle acceleration-range: the two other acceleration ranges in each benchmark correspond to generalization over distributional shifts.