Assumed Density Filtering and Smoothing with Neural Network Surrogate Models

Simon Kuang SLKU@UCDAVIS.EDU

Xinfan Lin LXFLIN@UCDAVIS.EDU

University of California, Davis

Editors: G. Sukhatme, L. Lindemann, S. Tu, A. Wierman, N. Atanasov

Abstract

The Kalman filter and Rauch-Tung-Striebel (RTS) smoother are optimal for state estimation in linear dynamic systems. With nonlinear systems, the challenge consists in how to propagate uncertainty through the state transitions and output function. For the case of a neural network model, we enable accurate uncertainty propagation using a recent state-of-the-art analytic formula for computing the mean and covariance of a deep neural network with Gaussian input. We argue that cross entropy is a more appropriate performance metric than RMSE for evaluating the accuracy of filters and smoothers. We demonstrate the superiority of our method for state estimation on a stochastic Lorenz system and a Wiener system, and find that our method enables more optimal linear quadratic regulation when the state estimate is used for feedback. Code available at https://github.com/simontheflutist/analytic-moments.

Keywords: filtering, smoothing, neural networks, uncertainty propagation

1. Introduction

Neural networks can model dynamic systems by virtue of their universal approximator property (Narendra and Parthasarathy, 1992; Masri et al., 1993). They can be supervised to fit trajectory data (Pillonetto et al., 2025) or "physics-informed" to satisfy known equations of motion (Mohajerin and Waslander, 2019; Michałowska et al., 2024). Beyond function approximation, there is deeper interest in analyzing what neural networks have to offer to the control theorist and vice versa, by finding analysis pathways in the compositional structure of an artificial neural network (Junnarkar et al., 2024; Xu and Sivaranjani, 2024).

This paper deals with the problem of filtering—state estimation from past and present output measurements. Classic applications include tracking and predicting the ballistic motion of a point mass on the basis of noisy measurements such as radar, or estimating one's own location by fusing inertial sensors and satellite navigation. Sundry applications include human physiology forecasting (Albers et al., 2023), pandemic surveillance (Alsaggaf et al., 2024), and latent macroeconomic variables (Burmeister and Wall, 1982). A restricted version of the filtering problem was solved in the '60s for linear systems with Gaussian process and measurement noise (Kalman and Bucy, 1961; Luenberger, 1964). For nonlinear systems, the theoretical Bayesian filter is intractable. The playing field for how to generalize the Kalman filter is wide and highly contested, and there are legion variational approximations such as the Extended and Unscented Kalman filters which claim to handle nonlinearity more faithfully (Särkkä and Svensson, 2023; Jiang et al., 2025). This paper's scope is limited to filters whose only state consists of a mean and covariance (thus excluding particle filters, ensemble Kalman filters, and PDE-based methods).

Existing work on state estimation with neural network dynamic models has treated the network as a black box, embedding it inside a general-purpose nonlinear Kalman filter such as the Extended Kalman Filter (Oveissi et al., 2025) or the Unscented Kalman Filter (Anurag et al., 2025). (Also see Bai et al. (2023) for an ample bibliography.) The basic challenge to these methods is propagating a Gaussian uncertainty through the dynamic model, as these methods are all perturbative in the small-variance limit.

Our work, which constructs a bespoke analytic Kalman filter for neural network dynamics, begins by recognizing that a neural network is not just any smooth black-box function. All of the nonlinearity of a neural network is concentrated in its activation function, which can be chosen to facilitate closed-form moment propagation (Kuang and Lin, 2026). With the right activation function, the first two moments of a Normal distribution can be propagated exactly through a single layer of a feedforward neural network, and by extension, approximately through a deep feedforward neural network. Across a range of applications, this uncertainty propagation method dramatically outperforms linearized and unscented propagation. The importance of moment accuracy is highlighted by Deisenroth et al. (2009), which finds that analytic moment propagation improves Kalman filtering on models represented as Gaussian processes. Our work turns to models represented as neural networks.

Recursive Bayesian state estimation can be implemented using three fundamental operations on functions: coupling (forming the joint distribution of two functions applied to the same input), conditioning, and uncertainty propagation. This paper implements the coupling operation using the grammar of neural networks. It imposes a variational approximation to simplify conditioning using the Gaussian formula. It tests different options for the plug-in uncertainty propagation method, including linearized propagation, unscented propagation, and analytic propagation (introduced in Kuang and Lin (2026)).

We find that analytic moment propagation through neural networks enables state estimation in problems hitherto believed to be intractable. The literature exhibits a survivorship bias for nonlinear filtering and smoothing problems that are amenable to general-purpose nonlinear Kalman filters (as detailed in the Background section). Usually, this amounts to selecting a sampling time that is small enough that the discretized dynamics are amenable to perturbative uncertainty propagation methods such as linearization. For example, the Lorenz system is filtered using a discretization time of 0.001 (Nosrati et al., 2011), 0.005 (Dubois et al., 2020), or 0.01 (Oveissi et al., 2025). Our work extends the discretization time to 1, which is on the order of a full period of the chaotic attractor.

We do not impeach the Extended, Unscented etc. Kalman filters as unfit for the problems on which they have been applied. Rather, by applying the analytic moment propagation method of Kuang and Lin (2026) to a neural network representation of the dynamics, we unlock Kalman filtering on harder, more nonlinear problems. On highly nonlinear problems, our method improves upon existing Kalman filters in both accuracy and calibration. We show an improvement in the point prediction and estimate of the state (accuracy). Qualitatively, we observe that our estimator is able to stay synchronized with the chaotic Lorenz system. We demonstrate that confidence regions predicted by our method's state covariances have better coverage than existing Kalman filters (calibration). That is, 95% confidence regions for the state contain the true state roughly 95% of the time. This enables risk-aware decision making for optimally trading between safety and performance.

2. Notation

When $\sigma : \mathbb{R} \to \mathbb{R}$ is a neural network activation function, $\sigma(x)$ for $x \in \mathbb{R}^n$ is applied elementwise. If X is a square-integrable random vector, the notation $\mathrm{N}\, X$ refers to a random variable having distribution $\mathcal{N}(\mathbb{E}\, X, \operatorname{Cov} X)$. If f is a neural network, the notation $\mathrm{N}^*\, f(X)$ is a Normal random variable in which the normality approximation is applied layer-by-layer as in the analytic propagation method of Kuang and Lin (2026). The comma , is a higher-order function: if $x \mapsto f(x)$ and $x \mapsto g(x)$ are functions, then (f, g) is the function $x \mapsto (f(x), g(x))$.

3. Problem statement

A dynamic system is described by

$$x_0 \sim \mathcal{N}(\mu_0, \Sigma_0) \tag{1a}$$

$$x_t = F(x_{t-1}, u_t) + \eta_t, \qquad \eta_t \sim \mathcal{N}(0, Q) \qquad \forall t \in \{1 \dots T\} \tag{1b}$$

$$y_t = H(x_t, u_t) + \epsilon_t, \qquad \epsilon_t \sim \mathcal{N}(0, R) \qquad \forall t \in \{1 \dots T\} \tag{1c}$$

where $x_t \in \mathbb{R}^{n_x}$ is the state, $u_t \in \mathbb{R}^{n_u}$ is the input, and $y_t \in \mathbb{R}^{n_y}$ is the output. The random variables x_0, $\{\eta_t\}_{t=1}^T$, and $\{\epsilon_t\}_{t=1}^T$ are independent.

This model motivates three problems.

Problem 1 (Prediction) *Given $\{u_s\}_{s=0}^{t-1}$ and $\{y_s\}_{s=1}^{t-1}$, predict x_t and y_t as a joint distribution $(\hat{X}_{t|t-1}, \hat{Y}_{t|t-1})$.*

Problem 2 (Filtering) *Given $\{u_s\}_{s=0}^{t}$ and $\{y_s\}_{s=1}^{t}$, estimate $\hat{X}_{t|t}$.*

Problem 3 (Smoothing) *Given $\{u_s\}_{s=0}^{T}$ and $\{y_s\}_{s=1}^{T}$, estimate $\hat{X}_{t|T}$.*

In the case that F and H are linear functions, the predictive and posterior distributions arising in these problems are Normal and can be computed analytically by recursion across time steps. However, in our case, F and H are nonlinear functions. Pursuant to the Assumed Density Filtering Ansatz (Deisenroth et al., 2009), we impose Normality assumptions on nonlinearly transformed Normal variables.

Abstract Kalman filtering The Kalman filter (Algorithm 1) solves the prediction and filtering problems by a forward recursion. In `Predict`, it uses the estimate of the current state X to compute the jointly Gaussian distribution (X', Y') of the next state and next output.[1] In `Update`, it incorporates the measurement y by conditioning the joint predictive distribution (X', Y') on $Y = y$. We also benchmark against an optional refinement `Recal` developed in Jiang et al. (2025, 2026), listed in Algorithm 2, which recalibrates the corrected state covariance using the posterior mean, in cases in which H is a nonlinear function.

1. Frequently (e.g. in Jiang et al. 2025) this step is notated in math and implemented in code as three separate uncertainty propagations: first, the uncertainty of X is propagated through the dynamic model to get X'; second, the uncertainty of X' is propagated through the observation model to get Y'; third, the uncertainty of X' is propagated through the observation model again to compute the cross-covariance of X' and Y'. Our method combines the latter two steps by using a neural network to express the coupling of X' and Y'. This enables the analytic implementation of the neural network uncertainty propagation operator developed in Kuang and Lin (2026).

Abstract RTS smoothing The Rauch-Tung-Striebel smoother Algorithm 3 solves the smoothing problem by a backward recursion. Algorithm 3 is also implemented as `Predict` and `Update`. In `Predict`, it uses the estimate of the current state X to compute the jointly Gaussian distribution (X, X') of the current and next state. In `Update`, it uses the estimate of the next state X' to update the estimate of the current state X.

In Appendix A, we list the pseudocode for these algorithms using high-level probabilistic notation similar to Hennig et al. (2022, Algorithms 5.3–4, 38.1–2), in which the principal objects are probability distributions represented by capital letters.

4. Neural networks

We introduce a preferred formalism for neural networks. The four-parameter (A, b, C, d) layer function is more general than customary. This generality allows for certain possibilities such as residual networks ($C = I$, $d = 0$), as well as computing certain joint distributions of crucial importance in Kalman filtering and RTS smoothing.

Definition 1 *A layer function is a function* $g : \mathbb{R}^n \times \mathbb{R}^{m\times n} \times \mathbb{R}^m \times \mathbb{R}^{m\times n} \times \mathbb{R}^m \to \mathbb{R}^m$ *defined by* $g(x; A, b, C, d) = \sigma(Ax + b) + Cx + d$*, where* $A \in \mathbb{R}^{m\times n}, b \in \mathbb{R}^m, C \in \mathbb{R}^{m\times n}, d \in \mathbb{R}^m$ *are parameters.*

Definition 2 *A neural network with* ℓ *layers is the function* $f : \mathbb{R}^{n_x} \to \mathbb{R}^{n_y}$ *defined by*

$$\begin{aligned} f(x) &= f^\ell(x) \\ f^k(x) &= g(f^{k-1}(x); A^k, b^k, C^k, d^k) && \forall k \in \{1 \dots \ell\} \\ f^0(x) &= x \end{aligned}$$

4.1. The identity-augmentation operator

Algorithm 1 requires the joint distribution of (X_t, Y_t), and Algorithm 3 requires the joint distribution of (X_t, X_{t+1}). Both of these distributions can be computed using the identity-augmentation operator $F \mapsto F_{\text{aug}} = (\text{id}, F)$. We construct a representation of F_{aug} that is itself a neural network. This construction is not inherently profound, but becomes extremely useful (when paired with a general-purpose Normal approximation of a neural network's output) as a mathematical and programmatic way of computing the joint distribution of a neural network's input and its output.

Lemma 3 *Neural networks defined by Def. 2 are closed under the input coupling: if* f_1 *and* f_2 *are two neural networks with* n *inputs and* ℓ *layers, then* (f_1, f_2) *can also be parameterized by a neural network with* n *inputs and* ℓ *layers.*

The idea of the construction, detailed in Appendix C, is to build block-identity C matrices. The first layer repeats the input by $x \mapsto (x, x)$, and the rest of the network is a direct sum of the layers of f_1 and f_2.

Corollary 4 *If* f *is a neural network with* n *inputs and* ℓ *layers, then* (id, f) *can be represented by a neural network with* n *inputs and* ℓ *layers.*

Proof In order to appeal to Lemma 3, we just have to represent the identity map as a neural network with ℓ layers. This can be done by setting $A^k = 0$, $b^k = 0$, $C^k = I$ and $d^k = 0$ for all $k \in \{1 \dots \ell\}$. ■

4.2. Methods for uncertainty propagation: with application to Kalman filtering

Let X be a multivariate Normal random variable and F a neural network following Def. 2. Uncertainty propagation refers to approximating the mean and covariance matrix of $Y = F(X)$. It appears as the "N" operator in the `Predict` step of Alg. 1. The introductory material draws on (Kuang and Lin, 2026, §3).

For any σ, the exact moments of a single layer are given by three transcendental functions $M_\sigma, K_\sigma, L_\sigma$ corresponding to bivariate Gaussian integrals:

Lemma 5 (Lemma 1, Kuang and Lin (2026)) *For some activation function σ, let g_σ be the function defined by $g_\sigma(x; A, b, C, d) = \sigma(Ax + b) + Cx + d$. Let $X \sim \mathcal{N}(\mu, \Sigma)$. Then*

$$\big(\mathbb{E}\, g_\sigma(X; A, b, C, d)\big)_i = M_\sigma(\mu_i; \nu_{ii}) + (C\mu)_i + d_i$$

and

$$\begin{aligned}\big(\operatorname{Cov} g_\sigma(X; A, b, C, d)\big)_{i,j} = {} & K_\sigma\left(\mu_i, \mu_j; \nu_{ii}, \nu_{jj}, \nu_{ij}\right) \\ & + L_\sigma\left(\mu_i; \nu_{ii}, \nu_{jj}, \kappa_{ij}\right) \\ & + L_\sigma\left(\mu_j; \nu_{jj}, \nu_{ii}, \kappa_{ji}\right) \\ & + \tau_{ij}.\end{aligned}$$

where for all valid indices (i, j),

$$\begin{aligned}\mu_i &= (A\mu + b)_i & \tau_{ij} &= (C\Sigma C^\intercal)_{i,j} \\ \nu_{ij} &= (A\Sigma A^\intercal)_{i,j} & \kappa_{ij} &= (A\Sigma C^\intercal)_{i,j}\end{aligned}$$

In particular, this work uses the sine activation function (Sitzmann et al., 2020), for which the moment maps are given by

$$M_{\sin}(\mu; \nu) = e^{-\nu/2} \sin(\mu) \qquad \text{(Lem. G.1, Kuang and Lin (2026))}$$

$$\begin{aligned}&K_{\sin}(\mu_1, \mu_2; \nu_{11}, \nu_{22}, \nu_{12}) \\ &\quad = \frac{1}{2}\left[e^{-\frac{\nu_{11}+\nu_{22}}{2}+\nu_{12}} - e^{-\frac{\nu_{11}+\nu_{22}}{2}}\right]\cos(\mu_1 - \mu_2) \\ &\quad - \frac{1}{2}\left[e^{-\frac{\nu_{11}+\nu_{22}}{2}-\nu_{12}} - e^{-\frac{\nu_{11}+\nu_{22}}{2}}\right]\cos(\mu_1 + \mu_2)\end{aligned}$$

(Lem. G.2, Kuang and Lin (2026))

$$L_{\sin}(\mu_1, \mu_2; \nu_{11}, \nu_{22}, \nu_{12}) = \nu_{12} e^{-\nu_{11}/2} \cos(\mu_1) \qquad \text{(Lem. G.3, Kuang and Lin (2026))}$$

This paper introduces the **ANALYTIC Kalman filter**, in which "N" is implemented using the layer-by-layer moment matching method introduced in Kuang and Lin (2026), which provides analytical expressions for the cases where σ is a Normal CDF function or a sinusoid.

Definition 6 *Let f be a neural network with ℓ layers. Given $X \sim \mathcal{N}(\mu, \Sigma)$, the layer-wise Gaussian approximation of $f(X)$, denoted $Y_{\text{ana}} = \mathrm{N}^* f(X)$, is the random variable defined by*

$$\begin{aligned} Y_{\text{ana}} &= Y^\ell \\ Y^k &= \mathrm{N}\, g(Y^{k-1}; A^k, b^k, C^k, d^k) \qquad\qquad \forall k \in \{1 \dots \ell\} \\ Y^0 &= X \end{aligned}$$

Baseline methods are presented in Appendix D. Each of these methods is also tested with the **RECALIBRATE** variation in which the UPDATE step (Alg. 1) is replaced by the recalibrate/back-out procedure described in Jiang et al. (2026), which claims that the RECALIBRATE variation of nonlinear Kalman filtering confers a more conservative management of state uncertainty in the presence of strong nonlinearity.

5. Performance criteria: beyond RMSE

At each time step, Problems 1, 2, and 3 call for a (predictive or posterior) distribution $\hat{X}$ with mean $\hat{\mu}$ and covariance $\hat{\Sigma}$. We also know the ground truth state x at each time step. Thus the performance of the prediction, filtering, or smoothing $\hat{X}$ should be understood via the joint distribution of $(x, \hat{\mu}, \hat{\Sigma})$.

This section explores the choice of a criterion function $J(x, \mu, \Sigma)$ to be averaged over the joint distribution of $(x, \hat{\mu}, \hat{\Sigma})$. If the underlying dynamic system is ergodic, then the phase average is also the time average; the average-case performance of the Kalman filter is the same as the long-run performance.

Definition 7 (RMSE) *The root-mean-square error is defined by $J(x, \mu, \Sigma) = \|x - \mu\|_2$.*

Most works we review treat the RMSE as the only figure of merit in Kalman filtering and ignore Σ.[2] But it is frequently asserted that the distribution $\mathcal{N}(\mu, \Sigma)$ should be interpreted as a Bayesian predictive or posterior distribution for x. Then it is not enough to have the right answer μ; the algorithm should also have the right uncertainty Σ. This agreement is called *calibration*. Calibration can be understood as a generalization of betting income on the outcome of a binary event (Cover and Thomas, 2006, Example 6.1.1). Linear filtering is calibrated, and state uncertainty can be used e.g. to trade off between safety and performance in control applications (Kishida, 2024). On the other hand, calibration failures have been blamed for the tendency of conversational large language models to assert confidently incorrect facts (Kalai et al., 2025). Let us examine the implications of calibration.

First, the triples (x, μ, Σ) should be appear to have come from a hierarchical model in which $x \sim \mathcal{N}(\mu, \Sigma)$ where (μ, Σ) have the true marginal distribution of the latter two variables in (x, μ, Σ).

$$\mu, \Sigma \sim \text{true distribution} \tag{2a}$$

$$x \mid \mu, \Sigma \sim \mathcal{N}(\mu, \Sigma) \tag{2b}$$

The goodness-of-fit of this model is summarized by the log likelihood $\log p_{x \sim \mathcal{N}(\mu, \Sigma)}(x)$. This induces the cross entropy criterion, which is also used in Deisenroth et al. (2009).

Definition 8 (Cross entropy) *The cross entropy criterion is defined by*

$$J(x, \mu, \Sigma) = \frac{1}{2} \log \det \Sigma + \frac{1}{2} (x - \mu)^\intercal \Sigma^{-1} (x - \mu).$$

2. For example, in Jiang et al. (2026), Σ does not attain a statistical meaning; its only responsibility is to generate the exploration region for the subsequent state update.

It can also be understood as the statistically optimal trade-off between confidence (small Σ) and caution (large Σ). Holding Σ fixed, the cross entropy is minimized on average when $\mu \approx x$. Holding μ fixed, the cross entropy is minimized on average when $\Sigma \approx \mathbb{E}\,(x-\mu)(x-\mu)^\intercal$. This shows that a lower cross entropy combines point estimation error and uncertainty estimation error.

Second, model (2) also implies $(x-\mu)^\intercal\Sigma^{-1}(x-\mu)$ has the χ^2 distribution with n_x degrees of freedom. Let $F_{\chi^2(n_x)}$ be its cumulative distribution function. Then for all $\alpha \in [0,1]$, it holds that

$$\mathbb{P}\left[(x-\mu)^\intercal\Sigma^{-1}(x-\mu) \leq F^{-1}_{\chi^2(n_x)}(1-\alpha)\right] \geq 1-\alpha.$$

Thus $\{x \mid (x-\mu)^\intercal\Sigma^{-1}(x-\mu) \leq F^{-1}_{\chi^2(n_x)}(1-\alpha)\}$ is a $(1-\alpha)$-confidence set for x (Wasserman, 2004, §6.3.2), which motivates the coverage criterion.

Definition 9 (Coverage) *The $(1-\alpha)$-coverage criterion is defined by*

$$J(x,\mu,\Sigma) = \begin{cases} 1, & (x-\mu)^\intercal\Sigma^{-1}(x-\mu) \leq F^{-1}_{\chi^2(n_x)}(1-\alpha) \\ 0, & \textit{otherwise} \end{cases}$$

If $\alpha = 0.05$, then the expected value of the $(1-\alpha)$-coverage criterion is the long-run frequentist probability that a 95% confidence set for x traps the true value of x. We simultaneously seek to minimize the volume of these confidence sets.

Definition 10 (Coverage volume) *The $(1-\alpha)$-coverage volume criterion is defined by*

$$J(x,\mu,\Sigma) = \left[F^{-1}_{\chi^2(n_x)}(1-\alpha)\right]^{n_x/2} V_{n_x}\sqrt{\det\Sigma},$$

where V_{n_x} is the volume of an n_x-ball.

A filter can have a low RMSE and good coverage yet still be statistically suboptimal, as the examples in Appendix B show. The takeaway of this section, therefore, is that cross entropy has the last word on the validity of a statistical inference.

6. Example: stochastic Lorenz system estimation using a surrogate model

The Lorenz system is a reduced-order continuous-time model of atmospheric convection. Here, we consider the problem of estimating all three states (x^1, x^2, x^3) of the Lorenz system from a sequence of noisy and temporally sparse measurements of x^1. The system of interest is the sampled stochastic Lorenz system with a deterministic initial condition.

$$x_0 = (-8, 4, 27) \tag{3a}$$

$$x_t = F(x_{t-1}, B_t) \qquad \forall t \in \{1 \dots T\}\,. \tag{3b}$$

B_t is an independent Wiener process for each $t \in \{1 \dots T\}$. The transition function F is given by

$$F(x,B) = \xi(\Delta t) \tag{4}$$

subject to

$$\xi(0) = x \tag{5}$$

$$\xi(t) = \int_0^t f(\xi(s))\,\mathrm{d}s + \sigma_\eta B(t) \qquad \forall t \in [0, \Delta t] \tag{6}$$

where f is the Lorenz vector field. (See Appendix E for details.)

6.1. Results

The discussion rounds to three significant digits and refers to the full results in Appendix H.

The ANALYTIC method achieves the lowest RMSE at 15.2 (prediction), 9.76 (filtering), and 9.58 (smoothing) (Table 1). The next best method, UNSCENTED'95 (RECAL), has a prediction RMSE of 16.2, filtering RMSE of 12.5, and smoothing RMSE of 14.9. On an RMSE basis, from prediction to filtering to smoothing, LINEAR performs worse despite seeing more data (21.4, 71.1, 72.3). This shows that the linearized dynamics don't contain enough information to assimilate new data. Moreover, all of these RMSEs are greater than the population RMS variation of the Lorenz system; the EKF performs worse than doing nothing (ignoring the measurements and reporting the same state at every time step). The UNSCENTED'02 Kalman Filters performs yet an order of magnitude worse. Compared to UNSCENTED'95, the highly localized hyperparameter tuning of UNSCENTED'02 results in profound instability to the point of numerical indeterminacy (with covariance matrices having condition numbers on the order of 10^{14}).

We realized only after experiments that MEAN-FIELD uncertainty propagation is simply not eligible for recursive Bayesian inference at all, as its mean-field Ansatz nullifies the cross-covariance computations set forth in §4.1. The output of MEAN-FIELD ends up being a stationary distribution with no updates.

The ANALYTIC methods' coverage meets or slightly exceeds the nominal 95% at 98.5% (prediction), 97.9% (filtering), and 97.7% (smoothing) (Table 2). In contrast, LINEAR performs much worse with 5.9%, 6.1%, and 1.2% coverage, and UNSCENTED'95 achieves 50.5%, 53.5%, and 35.8% coverage respectively. The disastrous coverage of these two methods is explained by their smaller confidence regions: for example, the LINEAR smoother's 95% confidence regions are on average 1000x smaller than those of the ANALYTIC smoother (Table 3).

Finally, the cross entropy of ANALYTIC (10.1, 5.72, 5.67) is significantly lower than all of the others (Table 4).

Visual inspection of the trajectory and coverage plots (Figures 1–10) provides qualitative insights that complement these quantitative results. The trajectory plots superimpose 100 time steps of the ground truth on top of the method's 90% confidence region. The ANALYTIC method (Figure 1) exhibits the tightest alignment with the ground truth. In contrast, the LINEAR method (Figure 5) is confidently incorrect. The UNSCENTED variants (Figures 7–9) demonstrate intermediate performance.

The ANALYTIC method achieves close to nominal coverage across all confidence levels (Figure 2), whereas the LINEAR method exhibits drastic undercoverage at all confidence levels (Figure 6).

In summary, the ANALYTIC method results in truer point predictions and point estimates as well as truer confidence regions than all of the other methods, by up to a millionfold.

7. Example: LTI dynamics with nonlinear output

This section deals with the discrete-time Wiener system:

$$x_{t+1} = Ax_t + Bu_t + \eta_t \qquad \eta_t \sim \mathcal{N}(0, Q) \tag{7}$$

$$y_t = H(x_t, u_t) + \epsilon_t \qquad \epsilon_t \sim \mathcal{N}(0, R) \tag{8}$$

where $x_t \in \mathbb{R}^{n_x}$, $u_t \in \mathbb{R}^{n_u}$, $y_t \in \mathbb{R}^{n_y}$, $A \in \mathbb{R}^{n_x \times n_x}$, $B \in \mathbb{R}^{n_x \times n_u}$, $Q \in \mathbb{R}^{n_x \times n_x}$, $R \in \mathbb{R}^{n_y \times n_y}$, $H : \mathbb{R}^{n_x + n_u} \to \mathbb{R}^{n_y}$ is a known strongly nonlinear function represented as a neural network, and η_t and ϵ_t are independent noise terms.

7.1. Prediction, filtering, smoothing of a stable system

We consider the case where A is stable, u is a sinusoidal signal, and the objective is to estimate $\{x_t\}$. The details on this system and its simulation are given in Appendix F. Full results are listed in Appendix I.

In brief, the ANALYTIC method scores the lowest RMSE across prediction (1.38), filtering (1.31), and smoothing (0.994) (Table 5). In second place is UNSCENTED'95, which scores 6.56, 6.54, and 6.61, respectively. Without recalibration, the worst RMSE is achieved by LINEAR (31.5, 31.5, 32.0), with UNSCENTED'02 slightly better (24.4, 24.6, 25.6).

Calibration wise, without recalibration LINEAR scores the worst coverage, with 28.4% (prediction), 25.8% (filtering), and 15.4% (smoothing) (Table 6); despite drawing confidence regions comparable in size to those of ANALYTIC, it captures the true state far less often. The UNSCENTED'02 method is similarly overconfident, covering only 57.6%, 55.9%, and 45.5%. In light of the moderate point estimation error, the poor coverage of these methods is striking (Figures 13–14).

The ANALYTIC method results in well-calibrated 95% confidence regions in all three problems, with 93.8% actual coverage in smoothing (Table 6). The UNSCENTED'95 smoother achieves 73.7% true coverage but with smaller confidence volumes than ANALYTIC (Table 7).

Balancing these competing priorities with the cross entropy criterion, we see that ANALYTIC dominates among non-recalibrated methods, with cross entropies of -1.73, -0.47, and 0.11 (Table 8). The LINEAR, UNSCENTED'95, and UNSCENTED'02 methods produce catastrophic cross entropies on the order of 10^2–10^5, reflecting severe overconfidence.

Effect of covariance recalibration. We also evaluate each method under the covariance-recalibrated framework of Jiang et al. (2025), which is claimed to improve Kalman filtering with a nonlinear H by re-evaluating the covariance around the updated state. Figures 19–22 summarize how each metric changes between the non-recalibrated and recalibrated variants of each method.

For ANALYTIC, recalibration has a small effect on RMSE (e.g., $0.994 \to 0.900$ in smoothing), while coverage improves slightly from 94.3% to 95.8% in prediction and from 93.8% to 96.3% in smoothing (Figures 19 and 20). This is consistent with the theoretical prediction of Jiang et al. (2025) that recalibration becomes unnecessary when the uncertainty propagation is already accurate, since the re-approximated quantities then agree with the originals. In cross entropy, recalibration improves ANALYTIC's smoothing score from 0.11 to -4.64, yielding a well-separated first place (Fig. 22); the effect on prediction and filtering is smaller than ANALYTIC's first-place margin over the other methods. We do not have an explanation for this observation.

By contrast, recalibration produces huge improvements for the less accurate propagation methods. For UNSCENTED'95, it closes most of the gap to ANALYTIC: prediction RMSE improves from 6.56 to 1.68, coverage rises from 82.3% to 93.3%, and cross entropy drops from 918 to -1.69. For UNSCENTED'02, RMSE drops by nearly an order of magnitude (e.g., $24.4 \to 2.98$ in prediction), coverage jumps from 57.6% to 91.0%, and cross entropy falls from 1.1×10^4 to 0.097. For LINEAR, RMSE is cut by roughly a third (e.g., $31.5 \to 9.83$ in prediction), coverage rises from 28.4% to 68.2%, and cross entropy drops by two orders of magnitude (e.g., $2.4 \times 10^4 \to 103$). These results confirm the central claim of Jiang et al. (2025): the conventional Kalman update systematically underestimates posterior covariance when the uncertainty propagation is approximate, and re-evaluating the covariance around the updated state substantially mitigates this overconfidence.

The takeaway of this experiment is twofold. First, even when conventional methods can achieve a low RMSE, achieving statistical calibration takes more work, and ANALYTIC gives the best tradeoff

between sensitivity and specificity in its uncertainty estimates. Second, the benefits of covariance recalibration are most pronounced for methods with less accurate uncertainty propagation; when combined with the accurate moment propagation of ANALYTIC the benefits are less consistent, but can still yield best-in-class results.

7.2. Quadratic regulation of an unstable system

We consider the case where A is marginally stable, and the objective is to minimize a quadratic cost function of $\{x_t\}$ and $\{u_t\}$. The control law is the linear quadratic regulator $u = -K\hat{x}$, where $\hat{x}$ is the Kalman filter estimate. The details on this system and its simulation are given in Appendix G. Full results are listed in Appendix J.

Without recalibration, the total quadratic cost incurred by the ANALYTIC-powered controller is only 6% higher than that of the optimal linear quadratic regulator using linear state feedback (Table 9). All of the other state estimation methods (LINEAR, UNSCENTED'95, UNSCENTED'02) result in an effectively unstable closed-loop system, with costs of 8 500–31 000 times the optimal LQR. This degradation is explained by inferior state estimation: these filters exhibit 400–680 times the RMSE of ANALYTIC (Table 10), and their 95% confidence regions cover the true state less than 2% of the time, compared to 90.4% for ANALYTIC (Table 11).

Effect of covariance recalibration. For ANALYTIC, recalibration has negligible effect on the regulation cost (1.06× LQR with or without), consistent with the estimation results above.

For the other methods, recalibration has a striking but mixed effect on closed-loop performance. UNSCENTED'95 (RECAL) sees the most dramatic improvement: total cost drops from 10 200× to 10.2× LQR, RMSE falls from 16.6 to 0.49, and coverage rises from 1.7% to 85.2%. UNSCENTED'02 (RECAL) also improves substantially, with cost dropping from 8 500× to 342× LQR, RMSE from 14.9 to 2.96, and coverage from 1.7% to 94.6%. By contrast, LINEAR (RECAL) does not benefit: its total cost actually increases from 31 000× to 220 000× LQR, suggesting that the linearization errors in LINEAR are too severe for recalibration alone to rescue the closed-loop system.

8. Conclusion

The novelty of this work is that it applies analytic neural network uncertainty propagation to Kalman filtering and RTS smoothing by explicitly constructing input couplings as neural networks. We view the different Kalman filter variants as induced by different versions of uncertainty propagation, which become interchangable implementations of the same abstract Kalman filter. Also, this work raises the issue of calibration in nonlinear Kalman filtering and RTS smoothing, which is needed to ensure that the posterior distributions' Gaussian approximations are valid.

The significance of this work is that state estimation powered by neural network uncertainty propagation delivers more accurate state estimates, with better-calibrated uncertainty, than existing methods. Our numerical examples exemplify this dramatic improvement by not only achieving state-of-the-art performance on a challenging Lorenz system identification problem, but also by delivering more accurate state estimates than existing methods on a linear system estimation problem. In applications, this means more nonlinear systems, longer sampling periods between measurements, greater process and observation noise, weaker measurements, more optimal risk-aware decisions, and more optimal state-feedback controllers.

Acknowledgments

This work was supported by the National Science Foundation CAREER Program (Grant No. 2046292).

References

Abdullah Akgül, Manuel Haußmann, and Melih Kandemir. Deterministic Uncertainty Propagation for Improved Model-Based Offline Reinforcement Learning, January 2025. URL http://arxiv.org/abs/2406.04088. arXiv:2406.04088 [cs].

David Albers, Melike Sirlanci, Matthew Levine, Jan Claassen, Caroline Der Nigoghossian, and George Hripcsak. Interpretable Forecasting of Physiology in the ICU Using Constrained Data Assimilation and Electronic Health Record Data, May 2023. URL http://arxiv.org/abs/2305.06513. arXiv:2305.06513 [q-bio, stat].

Abdulrahman U. Alsaggaf, Maryam Saberi, Tyrus Berry, and Donald Ebeigbe. Nonlinear Kalman Filtering in the Absence of Direct Functional Relationships Between Measurement and State. *IEEE Control Systems Letters*, 8:2865–2870, 2024. ISSN 2475-1456. doi:10.1109/LCSYS.2024.3514818. URL https://ieeexplore.ieee.org/document/10787252.

Kumar Anurag, Kasra Azizi, Francesco Sorrentino, and Wenbin Wan. RCUKF: Data-Driven Modeling Meets Bayesian Estimation, August 2025. URL http://arxiv.org/abs/2508.04985. arXiv:2508.04985 [cs].

Yuting Bai, Bin Yan, Chenguang Zhou, Tingli Su, and Xuebo Jin. State of art on state estimation: Kalman filter driven by machine learning. *Annual Reviews in Control*, 56:100909, January 2023. ISSN 1367-5788. doi:10.1016/j.arcontrol.2023.100909. URL https://www.sciencedirect.com/science/article/pii/S1367578823000731.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

Edwin Burmeister and Kent D. Wall. Kalman filtering estimation of unobserved rational expectations with an application to the German hyperinflation. *Journal of Econometrics*, 20 (2):255–284, November 1982. ISSN 03044076. doi:10.1016/0304-4076(82)90021-5. URL https://linkinghub.elsevier.com/retrieve/pii/0304407682900215.

Thomas M. Cover and Joy A. Thomas. *Elements of information theory*. Wiley-Interscience, Hoboken, N.J, second edition edition, 2006. ISBN 978-0-471-74882-3 978-0-471-74881-6 978-1-118-58577-1 978-0-471-24195-9. doi:10.1002/047174882X.

DeepMind, Igor Babuschkin, Kate Baumli, Alison Bell, Surya Bhupatiraju, Jake Bruce, Peter Buchlovsky, David Budden, Trevor Cai, Aidan Clark, Ivo Danihelka, Antoine Dedieu, Claudio Fantacci, Jonathan Godwin, Chris Jones, Ross Hemsley, Tom Hennigan, Matteo Hessel, Shaobo Hou, Steven Kapturowski, Thomas Keck, Iurii Kemaev, Michael King, Markus Kunesch, Lena

Martens, Hamza Merzic, Vladimir Mikulik, Tamara Norman, George Papamakarios, John Quan, Roman Ring, Francisco Ruiz, Alvaro Sanchez, Laurent Sartran, Rosalia Schneider, Eren Sezener, Stephen Spencer, Srivatsan Srinivasan, Miloš Stanojević, Wojciech Stokowiec, Luyu Wang, Guangyao Zhou, and Fabio Viola. The DeepMind JAX Ecosystem, 2020. URL http://github.com/google-deepmind.

Marc Peter Deisenroth, Marco F. Huber, and Uwe D. Hanebeck. Analytic moment-based Gaussian process filtering. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 225–232, Montreal Quebec Canada, June 2009. ACM. ISBN 978-1-60558-516-1. doi:10.1145/1553374.1553403. URL https://dl.acm.org/doi/10.1145/1553374.1553403.

Pierre Dubois, Thomas Gomez, Laurent Planckaert, and Laurent Perret. Data-driven predictions of the Lorenz system. *Physica D: Nonlinear Phenomena*, 408:132495, July 2020. ISSN 01672789. doi:10.1016/j.physd.2020.132495. URL https://linkinghub.elsevier.com/retrieve/pii/S0167278919307080.

James Foster, Goncalo dos Reis, and Calum Strange. High order splitting methods for SDEs satisfying a commutativity condition. *arXiv:2210.17543*, 2023.

Philipp Hennig, Michael A. Osborne, and Hans P. Kersting. *Probabilistic Numerics: Computation as Machine Learning*. Cambridge University Press, 1 edition, June 2022. ISBN 978-1-316-68141-1 978-1-107-16344-7. doi:10.1017/9781316681411. URL https://www.cambridge.org/core/product/identifier/9781316681411/type/book.

Marco F. Huber. Bayesian Perceptron: Towards fully Bayesian Neural Networks. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 3179–3186, December 2020. doi:10.1109/CDC42340.2020.9303764. URL https://ieeexplore.ieee.org/abstract/document/9303764. ISSN: 2576-2370.

Shida Jiang, Junzhe Shi, and Scott Moura. A New Framework for Nonlinear Kalman Filters, February 2025. URL http://arxiv.org/abs/2407.05717. arXiv:2407.05717 [eess].

Shida Jiang, Junzhe Shi, and Scott Moura. Mitigating Overconfidence in Nonlinear Kalman Filters via Covariance Recalibration, April 2026. URL http://arxiv.org/abs/2407.05717. arXiv:2407.05717 [eess].

S. Julier, J. Uhlmann, and H.F. Durrant-Whyte. A new method for the nonlinear transformation of means and covariances in filters and estimators. *IEEE Transactions on Automatic Control*, 45(3): 477–482, March 2000. ISSN 0018-9286. doi:10.1109/9.847726. URL http://ieeexplore.ieee.org/document/847726/. Publisher: Institute of Electrical and Electronics Engineers (IEEE).

Simon J. Julier and Jeffrey K. Uhlmann. New extension of the Kalman filter to nonlinear systems. In *Defense, Security, and Sensing*, 1997. URL https://api.semanticscholar.org/CorpusID:7937456.

S.J. Julier. The scaled unscented transformation. In *Proceedings of the 2002 American Control Conference*, volume 6, pages 4555–4559 vol.6, May 2002. doi:10.1109/ACC.2002.1025369. URL https://ieeexplore.ieee.org/document/1025369. ISSN: 0743-1619.

S.J. Julier, J.K. Uhlmann, and H.F. Durrant-Whyte. A new approach for filtering nonlinear systems. In *Proceedings of 1995 American Control Conference - ACC'95*, volume 3, pages 1628–1632, Seattle, WA, USA, 1995. American Autom Control Council. doi:10.1109/acc.1995.529783. URL http://ieeexplore.ieee.org/document/529783/.

Paul Jungmann, Julia Poray, and Akash Kumar. Analytical Uncertainty Propagation in Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems*, 36(2):2495–2508, February 2025. ISSN 2162-2388. doi:10.1109/TNNLS.2023.3347156. URL https://ieeexplore.ieee.org/document/10398277.

Neelay Junnarkar, Murat Arcak, and Peter Seiler. Synthesizing Neural Network Controllers with Closed-Loop Dissipativity Guarantees, April 2024. URL http://arxiv.org/abs/2404.07373. arXiv:2404.07373 [cs, eess].

Adam Tauman Kalai, Ofir Nachum, Santosh S. Vempala, and Edwin Zhang. Why Language Models Hallucinate, September 2025. URL http://arxiv.org/abs/2509.04664. arXiv:2509.04664 [cs].

R. E. Kalman and R. S. Bucy. New Results in Linear Filtering and Prediction Theory. *Journal of Basic Engineering*, 83(1):95–108, March 1961. ISSN 0021-9223. doi:10.1115/1.3658902. URL https://doi.org/10.1115/1.3658902.

Patrick Kidger. *On Neural Differential Equations*. PhD thesis, arXiv, February 2022. URL http://arxiv.org/abs/2202.02435. arXiv:2202.02435 [cs, math, stat].

Patrick Kidger and Cristian Garcia. Equinox: neural networks in JAX via callable PyTrees and filtered transformations. *Differentiable Programming workshop at Neural Information Processing Systems 2021*, 2021.

Masako Kishida. Risk-Aware Control of Discrete-Time Stochastic Systems: Integrating Kalman Filter and Worst-case CVaR in Control Barrier Functions. In *2024 IEEE 63rd Conference on Decision and Control (CDC)*, pages 2019–2024, December 2024. doi:10.1109/CDC56724.2024.10886199. URL https://ieeexplore.ieee.org/document/10886199/. ISSN: 2576-2370.

Simon Kuang and Xinfan Lin. Exact gaussian moment matching for residual networks: a second-order method, 2026. URL https://arxiv.org/abs/2601.22307.

Lennart Ljung. unscentedKalmanFilter. In *System Identification Toolbox Reference*, pages 1–2372. The MathWorks, Inc., 2025.

Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

David G. Luenberger. Observing the State of a Linear System. *IEEE Transactions on Military Electronics*, 8(2):74–80, April 1964. ISSN 2374-9520. doi:10.1109/TME.1964.4323124. URL https://ieeexplore.ieee.org/abstract/document/4323124.

S. F. Masri, A. G. Chassiakos, and T. K. Caughey. Identification of Nonlinear Dynamic Systems Using Neural Networks. *Journal of Applied Mechanics*, 60(1):123–133, March 1993. ISSN 0021-8936. doi:10.1115/1.2900734. URL https://doi.org/10.1115/1.2900734.

Katarzyna Michałowska, Somdatta Goswami, George Em Karniadakis, and Signe Riemer-Sørensen. Neural Operator Learning for Long-Time Integration in Dynamical Systems with Recurrent Neural Networks. In *2024 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, June 2024. doi:10.1109/IJCNN60899.2024.10650331. URL https://ieeexplore.ieee.org/abstract/document/10650331. ISSN: 2161-4407.

Nima Mohajerin and Steven L. Waslander. Multistep Prediction of Dynamic Systems With Recurrent Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems*, 30(11):3370–3383, November 2019. ISSN 2162-2388. doi:10.1109/TNNLS.2019.2891257. URL https://ieeexplore.ieee.org/abstract/document/8630673.

Tobias Nagel and Marco F. Huber. Kalman-Bucy-Informed Neural Network for System Identification. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 1503–1508, December 2022. doi:10.1109/CDC51059.2022.9993245. URL https://ieeexplore.ieee.org/document/9993245/. ISSN: 2576-2370.

Kumpati S. Narendra and Kannan Parthasarathy. Neural networks and dynamical systems. *International Journal of Approximate Reasoning*, 6(2):109–131, February 1992. ISSN 0888613X. doi:10.1016/0888-613X(92)90014-Q. URL https://linkinghub.elsevier.com/retrieve/pii/0888613X9290014Q.

K. Nosrati, A. Azemi, N. Pariz, and A. Shokouhi-R. Chaotic synchronization of Lorenz system using Unscented Kalman Filter. In *2011 Chinese Control and Decision Conference (CCDC)*, pages 848–853, May 2011. doi:10.1109/CCDC.2011.5968301. URL https://ieeexplore.ieee.org/document/5968301. ISSN: 1948-9447.

Parham Oveissi, Turibius Rozario, and Ankit Goel. A Novel Neural Filter to Improve Accuracy of Neural Network Models of Dynamic Systems, June 2025. URL http://arxiv.org/abs/2409.13654. arXiv:2409.13654 [cs].

Felix Petersen, Aashwin Mishra, Hilde Kuehne, Christian Borgelt, Oliver Deussen, and Mikhail Yurochkin. Uncertainty Quantification via Stable Distribution Propagation, February 2024. URL http://arxiv.org/abs/2402.08324. arXiv:2402.08324 [cs].

Gianluigi Pillonetto, Aleksandr Aravkin, Daniel Gedon, Lennart Ljung, Antônio H. Ribeiro, and Thomas B. Schön. Deep networks for system identification: A survey. *Automatica*, 171:111907, January 2025. ISSN 0005-1098. doi:10.1016/j.automatica.2024.111907. URL https://www.sciencedirect.com/science/article/pii/S0005109824004011.

Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit Neural Representations with Periodic Activation Functions. In *Advances in Neural Information Processing Systems*, volume 33, pages 7462–7473. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/hash/53c04118df112c13a8c34b38343b9c10-Abstract.html.

Simo Särkkä and Lennart Svensson. *Bayesian filtering and smoothing*. Institute of Mathematical Statistics textbooks. Cambridge University Press, New York, second edition edition, 2023. ISBN 978-1-108-92664-5.

Jessica S. Titensky, Hayden Jananthan, and Jeremy Kepner. Uncertainty Propagation in Deep Neural Networks Using Extended Kalman Filtering. In *2018 IEEE MIT Undergraduate Research Technology Conference (URTC)*, pages 1–4, October 2018. doi:10.1109/URTC45901.2018.9244804. URL https://ieeexplore.ieee.org/abstract/document/9244804.

Philipp Wagner, Xinyang Wu, and Marco F. Huber. Kalman Bayesian Neural Networks for Closed-form Online Learning, November 2022. URL http://arxiv.org/abs/2110.00944. arXiv:2110.00944 [cs].

E.A. Wan and R. Van Der Merwe. The unscented Kalman filter for nonlinear estimation. In *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No.00EX373)*, pages 153–158, Lake Louise, Alta., Canada, 2000. IEEE. ISBN 978-0-7803-5800-3. doi:10.1109/ASSPCC.2000.882463. URL http://ieeexplore.ieee.org/document/882463/.

Larry Wasserman. *All of Statistics: A Concise Course in Statistical Inference*. Springer Texts in Statistics. Springer, New York, NY, 2004. ISBN 978-1-4419-2322-6 978-0-387-21736-9. doi:10.1007/978-0-387-21736-9. URL http://link.springer.com/10.1007/978-0-387-21736-9.

Yuezhu Xu and S. Sivaranjani. ECLipsE: Efficient Compositional Lipschitz Constant Estimation for Deep Neural Networks. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 10414–10441. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/1419d8554191a65ea4f2d8e1057973e4-Paper-Conference.pdf.

Supplementary material

Contents

1 Introduction **1**

2 Notation **3**

3 Problem statement **3**

4 Neural networks **4**
4.1 The identity-augmentation operator 4
4.2 Methods for uncertainty propagation: with application to Kalman filtering 5

5 Performance criteria: beyond RMSE **6**

6 Example: stochastic Lorenz system estimation using a surrogate model **7**
6.1 Results 8

7 Example: LTI dynamics with nonlinear output **8**
7.1 Prediction, filtering, smoothing of a stable system 9
7.2 Quadratic regulation of an unstable system 10

8 Conclusion **10**

A Pseudocode for Kalman filter and RTS smoother **20**

B Motivating examples for confidence calibration **22**

C Proof of Lemma 3 **23**

D Filtering and Smoothing baseline methods **23**

E Supplement to §6 **25**

F Supplement to §7.1 **26**

G Supplement to §7.2 **27**

H Full results for §6 **28**

I Full results for §7.1 **36**
I.1 Effect of recalibration 43

J Full results for §7.2 **45**

K Runtime **48**

List of Tables

1 RMSE in the Lorenz system state estimation problem 28
2 Coverage at 95% in the Lorenz system state estimation problem 29
3 Volume at 95% in the Lorenz system state estimation problem 30
4 Cross entropy in the Lorenz system state estimation problem 31
5 RMSE for different methods and tasks in the LTI state estimation problem. 36
6 Coverage at 95% for different methods and tasks in the LTI state estimation problem. 37
7 Volume at 95% for different methods and tasks in the LTI state estimation problem. 38
8 Cross entropy for different methods and tasks in the LTI state estimation problem. . 39
9 LQR performance of different methods and tasks in the LTI regulation problem. . . 45
10 RMSE of the Kalman filters in closed loop in the LTI regulation problem. 46
11 Coverage at 95% of the Kalman filters in closed loop in the LTI regulation problem. 46
12 Volume at 95% of the Kalman filters in closed loop in the LTI regulation problem. . 47
13 Cross entropy of the Kalman filters in closed loop in the LTI regulation problem. . 47
14 Per-call runtime of Kalman and RTS primitives in the Lorenz state estimation problem. Wall clock reports mean ± standard error over 10 repetitions on CPU. 48
15 Per-call runtime of Kalman and RTS primitives in the LTI state-estimation problem. Wall clock reports mean ± standard error over 10 repetitions on CPU. 49

List of Figures

1 Trajectory excerpt for Kalman filter ANALYTIC in the Lorenz system state estimation problem. Plus sign indicates hit; cross indicates miss: 10 crosses is best for nominal coverage. . . . 29
2 Coverage for Kalman filter ANALYTIC in the Lorenz system state estimation problem. Closer to identity is best. . . . 30
3 Trajectory excerpt for Kalman filter MEAN-FIELD in the Lorenz system state estimation problem. Plus sign indicates hit; cross indicates miss: 10 crosses is best for nominal coverage. . . . 32
4 Coverage for Kalman filter MEAN-FIELD in the Lorenz system state estimation problem. Closer to identity is best. . . . 32
5 Trajectory excerpt for Kalman filter LINEAR in the Lorenz system state estimation problem. Plus sign indicates hit; cross indicates miss: 10 crosses is best for nominal coverage. . . . 33
6 Coverage for Kalman filter LINEAR in the Lorenz system state estimation problem. Closer to identity is best. . . . 33
7 Trajectory excerpt for Kalman filter UNSCENTED'95 in the Lorenz system state estimation problem. Plus sign indicates hit; cross indicates miss: 10 crosses is best for nominal coverage. . . . 34
8 Coverage for Kalman filter UNSCENTED'95 in the Lorenz system state estimation problem. Closer to identity is best. . . . 34
9 Trajectory excerpt for Kalman filter UNSCENTED'02 in the Lorenz system state estimation problem. Plus sign indicates hit; cross indicates miss: 10 crosses is best for nominal coverage. . . . 35
10 Coverage for Kalman filter UNSCENTED'02 in the Lorenz system state estimation problem. Closer to identity is best. . . . 35
11 Trajectory excerpt for Kalman filter ANALYTIC in the LTI state estimation problem. Plus sign indicates hit; cross indicates miss. Expect 10 misses for nominal 90% coverage. . . . 40
12 Trajectory excerpt for Kalman filter ANALYTIC (RECAL) in the LTI state estimation problem. Plus sign indicates hit; cross indicates miss. Expect 10 misses for nominal 90% coverage. . . . 40
13 Trajectory excerpt for Kalman filter LINEAR in the LTI state estimation problem. Plus sign indicates hit; cross indicates miss. Expect 10 misses for nominal 90% coverage. . . . 40
14 Trajectory excerpt for Kalman filter LINEAR (RECAL) in the LTI state estimation problem. Plus sign indicates hit; cross indicates miss. Expect 10 misses for nominal 90% coverage. . . . 41
15 Trajectory excerpt for Kalman filter UNSCENTED'95 in the LTI state estimation problem. Plus sign indicates hit; cross indicates miss. Expect 10 misses for nominal 90% coverage. . . . 41
16 Trajectory excerpt for Kalman filter UNSCENTED'95 (RECAL) in the LTI state estimation problem. Plus sign indicates hit; cross indicates miss. Expect 10 misses for nominal 90% coverage. . . . 41

17 Trajectory excerpt for Kalman filter UNSCENTED'02 in the LTI state estimation problem. Plus sign indicates hit; cross indicates miss. Expect 10 misses for nominal 90% coverage. 42
18 Trajectory excerpt for Kalman filter UNSCENTED'02 (RECAL) in the LTI state estimation problem. Plus sign indicates hit; cross indicates miss. Expect 10 misses for nominal 90% coverage. 42
19 Effect of recalibration on RMSE for each method and inference task in the LTI state estimation problem. Each line segment connects a method's no-recal value to its recal value; vertical axis is log-scaled. 43
20 Effect of recalibration on 95% coverage. 43
21 Effect of recalibration on 95% confidence volume (log-scaled). 44
22 Effect of recalibration on cross entropy. 44

Appendix A. Pseudocode for Kalman filter and RTS smoother

Algorithm 1: General Kalman algorithm for recursive **prediction** (problem 1) and **filtering** (problem 2)

Input : State transition function F and observation model H
Input : State covariance Q and observation covariance R
Input : Recalibrate flag `recal` $\in \{\texttt{true}, \texttt{false}\}$
Function `Predict` (X, u)
  `/* Propagate` X `through state transition */`
  $X' \leftarrow \mathrm{N}\, F(X, u) + \mathcal{N}(0, Q)$
  `/* Propagate` X' `through observation model */`
  $((X', u), Y') \leftarrow \mathrm{N}(\mathrm{id}, H)(X', u) + \mathcal{N}(0, 0_{n_x+n_u} \oplus R)$
  `/* Get joint distribution of next state and next output */`
  $(X', Y') \leftarrow$ projection of $((X', u), Y')$
  return (X', Y')
Function `Update` $((X, Y), y)$
  `/* Apply Bayes' rule */`
  $(X', y) \leftarrow$ conditional distribution of (X, Y) given $Y = y$
  return X'
Function `Filter` $(u_1, \ldots, u_T, y_1, \ldots, y_T)$
  $\hat{X}_{0|0} \leftarrow \mathcal{N}(\mu_0, \Sigma_0)$
  for $t \in [1 \ldots T]$ **do**
    `/* Solution to problem 1 */`
    $(\hat{X}_{t|t-1}, \hat{Y}_{t|t-1}) \leftarrow \texttt{Predict}(\hat{X}_{t-1|t-1}, u_t)$
    `/* Solution to problem 2 */`
    $\hat{X}_{t|t} \leftarrow \texttt{Update}((\hat{X}_{t|t-1}, \hat{Y}_{t|t-1}), y_t)$
    `/* Recalibration */`
    if *recal* **then** $\hat{X}_{t|t} \leftarrow \texttt{Recal}((\hat{X}_{t|t-1}, \hat{Y}_{t|t-1}), \hat{X}_{t|t}, y_t, u_t)$
  end
  return $\{\hat{X}_{t|t}\}_{t=1}^{T}$

Algorithm 2: Recalibrated update step generalized from Jiang et al. (2025)

Function `Recal` $((X, Y), X', y, u)$
  `/* Kalman prediction (Algorithm 1) using corrected mean */`
  $(X^+, Y^+) \leftarrow \texttt{Predict}(\mathcal{N}(\mathbb{E}\, X', \mathrm{Cov}\, X), u)$
  `/* Kalman gain */`
  $K \leftarrow \mathrm{Cov}(Y, X)\, \mathrm{Cov}(Y, Y)^{-1}$
  `/* Recalibrated covariance */`
  $$P^+ \leftarrow \begin{pmatrix} I \\ -K \end{pmatrix}^{\intercal} \begin{pmatrix} \mathrm{Cov}(X, X) & \mathrm{Cov}(X^+, Y^+) \\ \mathrm{Cov}(Y^+, X^+) & \mathrm{Cov}(Y^+, Y^+) \end{pmatrix} \begin{pmatrix} I \\ -K \end{pmatrix}$$
  `/* Back out if recalibration increased uncertainty */`
  if $\mathrm{tr}\, P^+ > \mathrm{tr}\, \mathrm{Cov}\, X$ **then return** $\mathcal{N}(\mathbb{E}\, X, \mathrm{Cov}\, X)$
  return $\mathcal{N}(\mathbb{E}\, X', P^+)$

Algorithm 3: General RTS algorithm for recursive **smoothing** (problem 3)

Input : State transition function F
Input : State covariance Q
Function `Predict` (X, u)
  `/* Propagate` X `through state transition */`
  $((X, u), X') \leftarrow \mathrm{N}(\mathrm{id}, F)(X, u) + \mathcal{N}(0, 0_{n_x+n_u} \oplus Q)$
  `/* Get joint distribution of current and next state */`
  return (X, X')
Function `Update` $((X, X'), X'')$
  `/* Apply Bayes' rule */`
  $(X, X'') \leftarrow$ conditional distribution of (X, X') given $X' = X''$
  return X
Function `Smooth` $(u_1, \ldots, u_T, y_1, \ldots, y_T)$
  $\{\hat{X}_{t|t}\}_{t=1}^{T} \leftarrow$ `Filter`$(u_1, \ldots, u_T, y_1, \ldots, y_T)$
  for $t \in [T-1, \ldots, 0]$ **do**
    `/* Compute joint distribution of current and next state */`
    $(\hat{X}_{t|t}, \hat{X}_{t+1|t}) \leftarrow$ `Predict`$(\hat{X}_{t|t}, u_{t+1})$
    `/* Solution to problem 3 */`
    $\hat{X}_{t|T} \leftarrow$ `Update`$((\hat{X}_{t|t}, \hat{X}_{t+1|t}), \hat{X}_{t+1|T})$
  end
  return $\{\hat{X}_{t|T}\}_{t=1}^{T}$

Appendix B. Motivating examples for confidence calibration

Let us consider three toy examples of joint distributions (x, μ, Σ) having a suboptimal Σ. We will write $J = J(\epsilon, \Sigma)$ with $\epsilon = x - \mu$. Every example has the same RMSE. We vary only the law of Σ and its coupling with ϵ in order to show that they make all the difference.

Example 1 *Let $\epsilon \sim \mathcal{N}(0, I)$ and $\Sigma = \frac{1}{2}I$. Then the 95% coverage criterion has expected value strictly less than 0.95.*

Example 2 *Let $\epsilon \sim \mathcal{N}(0, I)$ and $\Sigma = 2I$. Then the 95% coverage criterion has expected value strictly greater than 0.95.*

Example 3 *Let $\epsilon \sim \mathcal{N}(0, I)$ and, independently, Σ obeys*

$$\Sigma = \begin{cases} 0I & \text{with probability } 0.05 \\ \infty I & \text{with probability } 0.95 \end{cases}$$

Then the 95% coverage criterion has expectation 95%, which is perfect. But no other confidence level is correctly covered, and the coverage volume is too large. The cross entropy is $+\infty$.

This last example shows that coverage is necessary but not sufficient for a good predictive (posterior) distribution. We desire that x is trapped by Σ's confidence intervals not only in an average sense, but also conditional on x.

Example 4 *Consider the following mixture model. We flip an unfair coin that comes up heads with probability 0.05. If heads, then $\epsilon \sim \mathcal{N}(0, 1000I)$, $\Sigma_1 = 0.001I$, and $\Sigma_2 = 1000I$. If tails, then $\epsilon \sim \mathcal{N}(0, I)$, $\Sigma_1 = 1000I$, and $\Sigma_2 = I$.*

Both (ϵ, Σ_1) and (ϵ, Σ_2) approximately achieve 95% coverage. But the clearly superior Σ_2 minimizes the cross entropy criterion.

Appendix C. Proof of Lemma 3

For $j \in \{1, 2\}$, let f_j be defined by

$$f_j(x) = f_j^\ell(x), \tag{9}$$
$$f_j^k(x) = g(f_j^{k-1}(x); A_j^k, b_j^k, C_j^k, d_j^k), \qquad k \in \{1 \dots \ell\}, \tag{10}$$
$$f_j^0(x) = x \tag{11}$$

Now define $f_{\text{aug}} = (f_1, f_2)$ by

$$f_{\text{aug}}(x) = f_{\text{aug}}^\ell(x), \tag{12}$$
$$f_{\text{aug}}^k(x) = g(f_{\text{aug}}^{k-1}(x); A_{\text{aug}}^k, b_{\text{aug}}^k, C_{\text{aug}}^k, d_{\text{aug}}^k), \qquad k \in \{1 \dots \ell\}, \tag{13}$$
$$f_{\text{aug}}^0(x) = x \tag{14}$$

where

$$A_{\text{aug}}^1 = \begin{pmatrix} A_1 \\ A_2 \end{pmatrix} \qquad b_{\text{aug}}^1 = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} \tag{15}$$
$$C_{\text{aug}}^1 = \begin{pmatrix} C_1 \\ C_2 \end{pmatrix} \qquad d_{\text{aug}}^1 = \begin{pmatrix} d_1 \\ d_2 \end{pmatrix} \tag{16}$$

and for $k \in \{2 \dots \ell\}$,

$$A_{\text{aug}}^k = \begin{pmatrix} A_1^k & 0 \\ 0 & A_2^k \end{pmatrix} \qquad b_{\text{aug}}^k = \begin{pmatrix} b_1^k \\ b_2^k \end{pmatrix} \tag{17}$$
$$C_{\text{aug}}^k = \begin{pmatrix} C_1^k & 0 \\ 0 & C_2^k \end{pmatrix} \qquad d_{\text{aug}}^k = \begin{pmatrix} d_1^k \\ d_2^k \end{pmatrix} \tag{18}$$

Appendix D. Filtering and Smoothing baseline methods

The implementation of "N" is varied to generate baseline Kalman filters. Inspired by Huber (2020); Wagner et al. (2022); Akgül et al. (2025), we define a **MEAN-FIELD analytic Kalman filter** by assuming that neurons in the same hidden layer are independent:

$$\begin{aligned}
Y_{\text{mfa}} &= Y^\ell \\
Y^k &= \mathcal{N}(\mu^k, \Sigma^k) \quad \forall k \in \{1 \dots \ell\} \\
\mu^k &= \mathbb{E}\, g(Y^{k-1}; A^k, b^k, C^k, d^k) \\
\Sigma_{ij}^k &= \begin{cases} \left[\operatorname{Cov} g(Y^{k-1}; A^k, b^k, C^k, d^k)\right]_{ij}, & i = j \\ 0, & \text{else} \end{cases} \\
Y^0 &= X
\end{aligned}$$

The **EXTENDED Kalman filter** in works such as Jiang et al. (2026); Oveissi et al. (2025) uses a linearization of the neural network (Titensky et al., 2018; Nagel and Huber, 2022; Petersen et al.,

2024; Jungmann et al., 2025). The **UNSCENTED'95 Kalman filter** introduced in Julier et al. (1995); Julier and Uhlmann (1997); Julier et al. (2000) is a one-parameter family of transformations that approximate the distribution of X by $2n + 1$ point masses. The **UNSCENTED'02 Kalman filter** introduced in Julier (2002); Wan and Van Der Merwe (2000) adds two additional hyperparameters. It is more commonly used today (Jiang et al., 2025; Anurag et al., 2025) and is the default in off-the-shelf software (Ljung, 2025).

Appendix E. Supplement to §6

The Lorenz vector field is given by

$$f\begin{pmatrix} x^1 \\ x^2 \\ x^3 \end{pmatrix} = \begin{pmatrix} \sigma_L(x^2 - x^1) \\ x^1(\rho - x^3) - x^2 \\ x^1 x^2 - \beta x^3 \end{pmatrix} \tag{19a}$$

where the Lorenz parameters are $\sigma_L = 10$, $\rho = 28$, and $\beta = 8/3$. The process noise standard deviation is $\sigma_\eta = 0.001$. The measurement noise standard deviation is $\sigma_\epsilon = 0.1$. The sampling time is $\Delta t = 1.0$.

In order to learn the generative model 1 from a training set $\{x_t\}_{t=0}^{T_{\text{train}}}$, we learn the neural network F and process covariance Q simultaneously by maximizing the profile likelihood:

$$\begin{aligned} \min_{F,Q} \quad & \det Q \\ \text{subject to} \quad & Q = \frac{1}{T_{\text{train}} - 1} \sum_{t=2}^{T_{\text{train}}} \epsilon_t \epsilon_t^\intercal \\ & \epsilon_t = x_t - F(x_{t-1}, u_t) \end{aligned} \tag{20}$$

Implementation details

We generated data using the stochastic integrator (Foster et al., 2023) implemented by Diffrax (Kidger, 2022) within the JAX programming system (DeepMind et al., 2020; Kidger and Garcia, 2021; Bradbury et al., 2018). The training and validation datasets both had length $T_{\text{train}} = 200,000$. The neural network had five hidden layers with 64 units each. The activation function was sine, and there were residual connections between hidden layers.

We optimized (20) using 20,000 epochs of AdamW (Loshchilov and Hutter, 2019) implemented in Optax (DeepMind et al., 2020). We used a minibatch size of 10,000 and decayed the learning rate from 10^{-4} to 10^{-5} over 20,000 epochs. This took about four hours on a Nvidia T1200 GPU, an Intel® Core™ i7-11850H CPU, and 32GB of RAM.

We report mean $\pm$ standard error quantities based on twenty independent realizations of the test data.

Appendix F. Supplement to §7.1

We generated the dynamic system with the following parameters:

- State dimension: $n_x = 5$
- Output dimension: $n_y = 3$
- Input dimension: $n_u = 1$
- A and B are in controllable canonical form with eigenvalues $(0.9, 0.7, 0.5, 0.3, 0.1)$.
- H is a neural network with one hidden layer of 50 units and a Φ activation, followed by a linear layer. The weights and biases are randomly initialized from normal distributions with default scaling. The second linear layer has zero weights and zero biases. H is trained to interpolate $M = 100$ random input-output pairs using the CoCoB optimizer for 10,000 steps.
- Noise covariances:
  - Process noise: $Q = 10^{-3} I_{n_x}$
  - Measurement noise: $R = 10^{-3} I_{n_y}$

The input is $u(t) = \sin(0.2t)$. The initial state is 0 and the initial state estimate is $\mathcal{N}(0, 0)$.

The horizon is $T = 10000$ time steps. We report mean $\pm$ standard error quantities based on twenty independent realizations of the noises η_t and ϵ_t.

Appendix G. Supplement to §7.2

We generated the dynamic system with the following parameters:

- State dimension: $n_x = 4$
- Output dimension: $n_y = 8$
- Input dimension: $n_u = 1$
- A and B are in controllable canonical form with eigenvalues $(1, -1, 0.1, -0.1)$.
- H is a neural network with one hidden layer of 50 units and a Φ activation, follow by another affine layer and a Φ activation. The weights and biases are randomly initialized from normal distributions. In the first hidden layer, the weights are scaled by ten times the inverse square root of the number of inputs, and the biases are scaled by the inverse square root of the number of inputs. In the second hidden layer, the weights are scaled by the inverse square root of the number of inputs and the biases are set to zero.
- Noise covariances:
  - Process noise: $Q = 10^{-2} I_{n_x}$
  - Measurement noise: $R = 10^{-4} I_{n_y}$

The control objective is to minimize $\sum_t \|x_t\|^2 + \sum_t \|u_t\|^2$. The initial state is 0 and the initial state estimate is $\mathcal{N}(0, 0)$.

The horizon is $T = 10000$ time steps. We report mean $\pm$ standard error quantities based on twenty independent realizations of the noises η_t and ϵ_t.

Appendix H. Full results for §6

Table 1: RMSE in the Lorenz system state estimation problem

Method	Task	Value $\pm$ Monte Carlo standard error
ANALYTIC	Prediction $(t\|t-1)$	$+1.525 \times 10^{+1} \pm 1.1 \times 10^{-2}$
	Filtering $(t\|t)$	$+9.762 \times 10^{0} \pm 9.9 \times 10^{-3}$
	Smoothing $(t\|T)$	$+9.579 \times 10^{0} \pm 1.0 \times 10^{-2}$
MEAN-FIELD	Prediction $(t\|t-1)$	$+2.007 \times 10^{+1} \pm 2.2 \times 10^{-2}$
	Filtering $(t\|t)$	$+2.007 \times 10^{+1} \pm 2.2 \times 10^{-2}$
	Smoothing $(t\|T)$	$+2.007 \times 10^{+1} \pm 2.2 \times 10^{-2}$
LINEAR	Prediction $(t\|t-1)$	$+2.137 \times 10^{+1} \pm 2.9 \times 10^{-2}$
	Filtering $(t\|t)$	$+7.109 \times 10^{+1} \pm 2.3 \times 10^{0}$
	Smoothing $(t\|T)$	$+7.232 \times 10^{+1} \pm 2.3 \times 10^{0}$
UNSCENTED'95	Prediction $(t\|t-1)$	$+1.623 \times 10^{+1} \pm 2.3 \times 10^{-2}$
	Filtering $(t\|t)$	$+1.256 \times 10^{+1} \pm 4.3 \times 10^{-2}$
	Smoothing $(t\|T)$	$+1.507 \times 10^{+1} \pm 5.6 \times 10^{-2}$
UNSCENTED'02	Prediction $(t\|t-1)$	— $\pm$ —
	Filtering $(t\|t)$	— $\pm$ —
	Smoothing $(t\|T)$	— $\pm$ —

Table 2: Coverage at 95% in the Lorenz system state estimation problem

Method	Task	Value ± Monte Carlo standard error
ANALYTIC	Prediction $(t\|t-1)$	$+9.850 \times 10^{-1} \pm 2.2 \times 10^{-4}$
	Filtering $(t\|t)$	$+9.794 \times 10^{-1} \pm 2.1 \times 10^{-4}$
	Smoothing $(t\|T)$	$+9.767 \times 10^{-1} \pm 2.6 \times 10^{-4}$
MEAN-FIELD	Prediction $(t\|t-1)$	$+7.195 \times 10^{-1} \pm 1.3 \times 10^{-3}$
	Filtering $(t\|t)$	$+7.195 \times 10^{-1} \pm 1.3 \times 10^{-3}$
	Smoothing $(t\|T)$	$+7.195 \times 10^{-1} \pm 1.3 \times 10^{-3}$
LINEAR	Prediction $(t\|t-1)$	$+5.942 \times 10^{-2} \pm 1.1 \times 10^{-3}$
	Filtering $(t\|t)$	$+6.143 \times 10^{-2} \pm 1.1 \times 10^{-3}$
	Smoothing $(t\|T)$	$+1.223 \times 10^{-2} \pm 5.8 \times 10^{-4}$
UNSCENTED'95	Prediction $(t\|t-1)$	$+5.050 \times 10^{-1} \pm 1.3 \times 10^{-3}$
	Filtering $(t\|t)$	$+5.346 \times 10^{-1} \pm 1.3 \times 10^{-3}$
	Smoothing $(t\|T)$	$+2.728 \times 10^{-1} \pm 1.2 \times 10^{-3}$
UNSCENTED'02	Prediction $(t\|t-1)$	$+9.677 \times 10^{-1} \pm 2.5 \times 10^{-2}$
	Filtering $(t\|t)$	$+9.360 \times 10^{-1} \pm 2.4 \times 10^{-2}$
	Smoothing $(t\|T)$	$+8.491 \times 10^{-1} \pm 6.3 \times 10^{-2}$

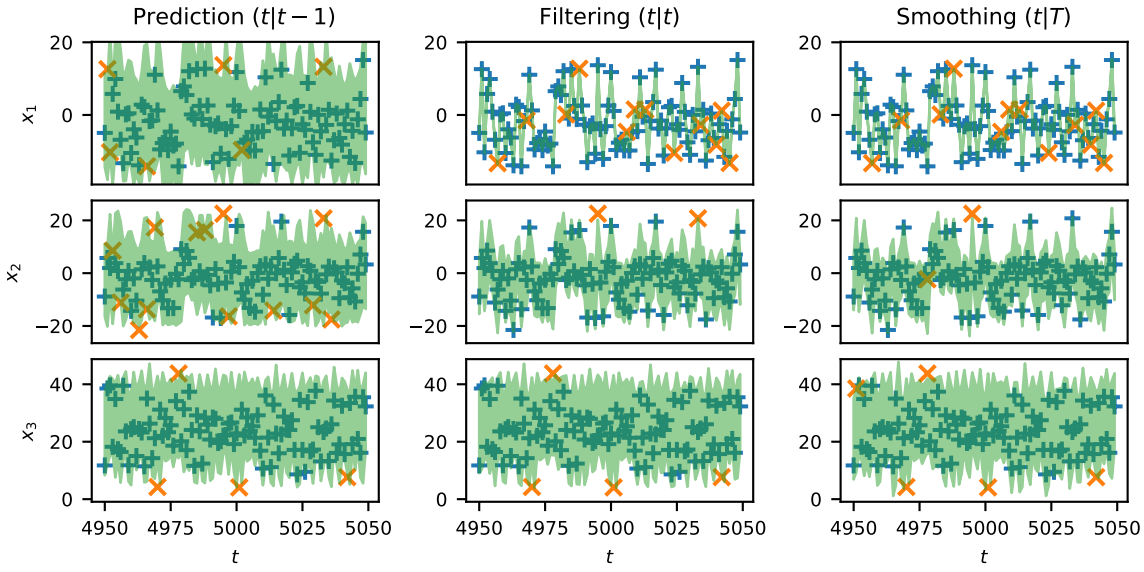

Figure 1: Trajectory excerpt for Kalman filter ANALYTIC in the Lorenz system state estimation problem. Plus sign indicates hit; cross indicates miss: 10 crosses is best for nominal coverage.

Table 3: Volume at 95% in the Lorenz system state estimation problem

Method	Task	Value ± Monte Carlo standard error
ANALYTIC	Prediction $(t\|t-1)$	$+4.609 \times 10^{+4} \pm 1.8 \times 10^{+1}$
	Filtering $(t\|t)$	$+5.053 \times 10^{+2} \pm 1.1 \times 10^{-1}$
	Smoothing $(t\|T)$	$+4.823 \times 10^{+2} \pm 1.2 \times 10^{-1}$
MEAN-FIELD	Prediction $(t\|t-1)$	$+4.984 \times 10^{+4} \pm 0$
	Filtering $(t\|t)$	$+4.984 \times 10^{+4} \pm 0$
	Smoothing $(t\|T)$	$+4.984 \times 10^{+4} \pm 0$
LINEAR	Prediction $(t\|t-1)$	$+7.236 \times 10^{+3} \pm 1.1 \times 10^{+2}$
	Filtering $(t\|t)$	$+4.040 \times 10^{+1} \pm 2.5 \times 10^{-1}$
	Smoothing $(t\|T)$	$+1.644 \times 10^{-1} \pm 9.3 \times 10^{-4}$
UNSCENTED'95	Prediction $(t\|t-1)$	$+9.029 \times 10^{+3} \pm 3.1 \times 10^{+1}$
	Filtering $(t\|t)$	$+1.349 \times 10^{+2} \pm 4.1 \times 10^{-1}$
	Smoothing $(t\|T)$	$+5.061 \times 10^{+1} \pm 1.8 \times 10^{-1}$
UNSCENTED'02	Prediction $(t\|t-1)$	— ± —
	Filtering $(t\|t)$	— ± —
	Smoothing $(t\|T)$	— ± —

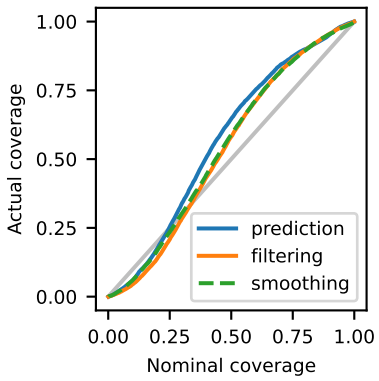

Figure 2: Coverage for Kalman filter ANALYTIC in the Lorenz system state estimation problem. Closer to identity is best.

Table 4: Cross entropy in the Lorenz system state estimation problem

Method	Task	Value $\pm$ Monte Carlo standard error
ANALYTIC	Prediction $(t\|t-1)$	$+1.014 \times 10^{+1} \pm 1.9 \times 10^{-3}$
	Filtering $(t\|t)$	$+5.717 \times 10^{0} \pm 1.2 \times 10^{-3}$
	Smoothing $(t\|T)$	$+5.667 \times 10^{0} \pm 1.4 \times 10^{-3}$
MEAN-FIELD	Prediction $(t\|t-1)$	$+1.223 \times 10^{+1} \pm 7.5 \times 10^{-3}$
	Filtering $(t\|t)$	$+1.223 \times 10^{+1} \pm 7.5 \times 10^{-3}$
	Smoothing $(t\|T)$	$+1.223 \times 10^{+1} \pm 7.5 \times 10^{-3}$
LINEAR	Prediction $(t\|t-1)$	$+3.088 \times 10^{+2} \pm 1.2 \times 10^{0}$
	Filtering $(t\|t)$	$+2.945 \times 10^{+2} \pm 1.2 \times 10^{0}$
	Smoothing $(t\|T)$	$+1.325 \times 10^{+6} \pm 1.2 \times 10^{+5}$
UNSCENTED'95	Prediction $(t\|t-1)$	$+2.391 \times 10^{+1} \pm 1.3 \times 10^{-1}$
	Filtering $(t\|t)$	$+1.824 \times 10^{+1} \pm 1.3 \times 10^{-1}$
	Smoothing $(t\|T)$	$+9.860 \times 10^{+1} \pm 1.7 \times 10^{0}$
UNSCENTED'02	Prediction $(t\|t-1)$	— $\pm$ —
	Filtering $(t\|t)$	— $\pm$ —
	Smoothing $(t\|T)$	— $\pm$ —

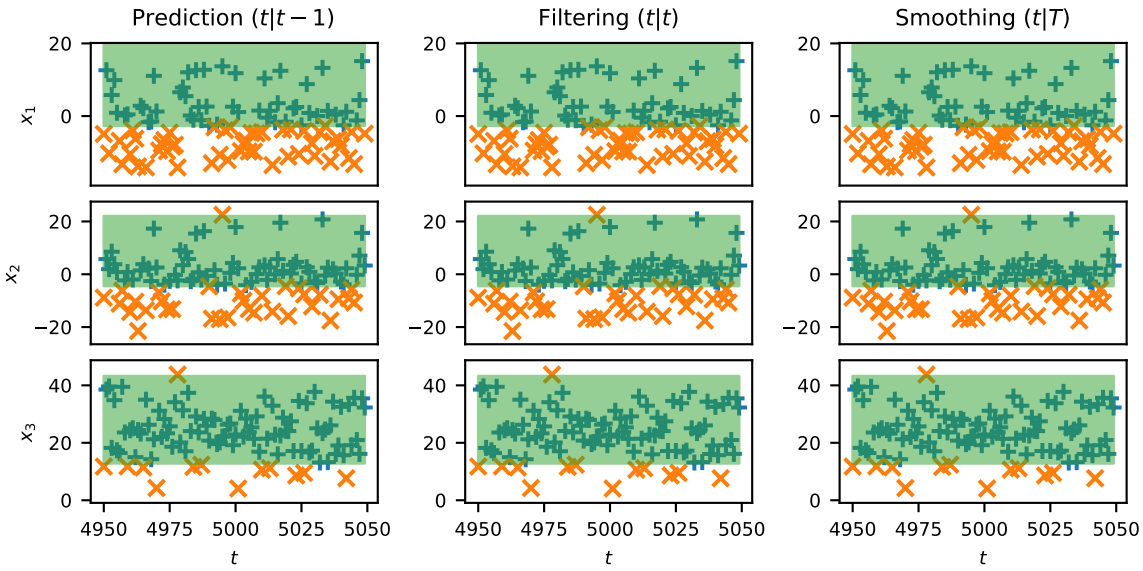

Figure 3: Trajectory excerpt for Kalman filter MEAN-FIELD in the Lorenz system state estimation problem. Plus sign indicates hit; cross indicates miss: 10 crosses is best for nominal coverage.

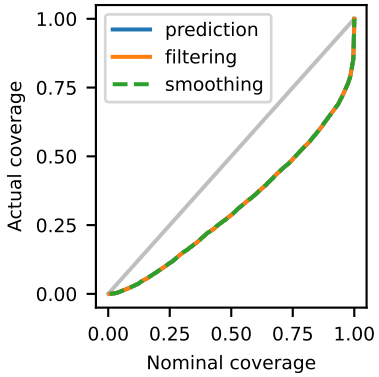

Figure 4: Coverage for Kalman filter MEAN-FIELD in the Lorenz system state estimation problem. Closer to identity is best.

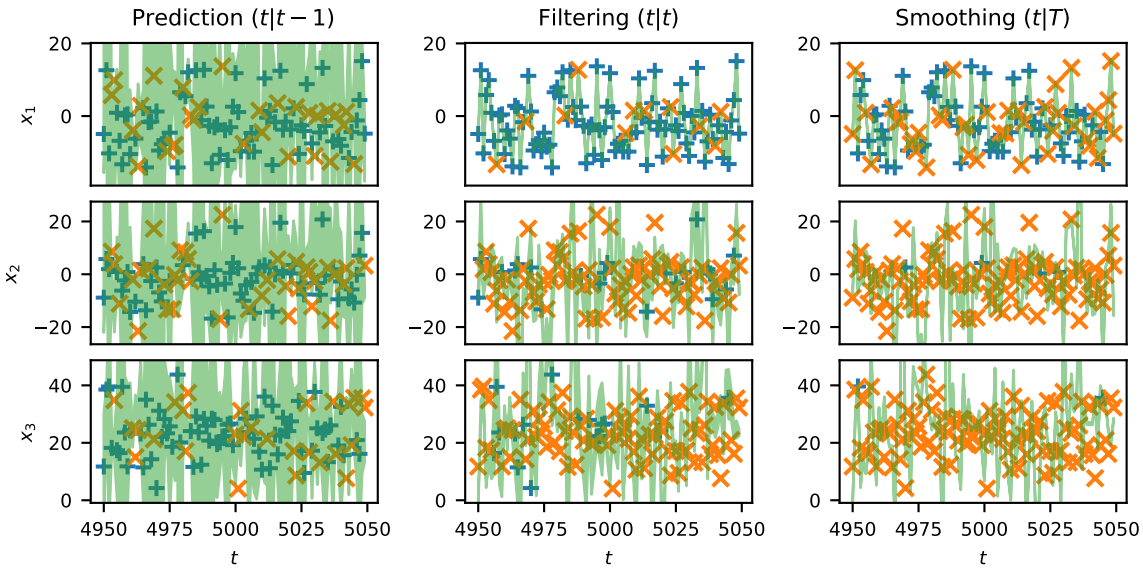

Figure 5: Trajectory excerpt for Kalman filter LINEAR in the Lorenz system state estimation problem. Plus sign indicates hit; cross indicates miss: 10 crosses is best for nominal coverage.

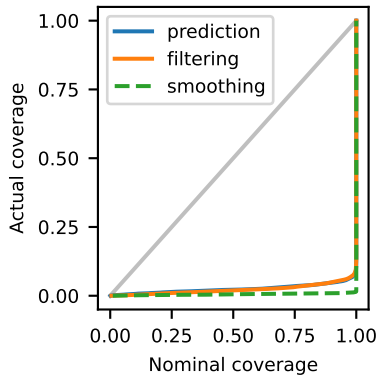

Figure 6: Coverage for Kalman filter LINEAR in the Lorenz system state estimation problem. Closer to identity is best.

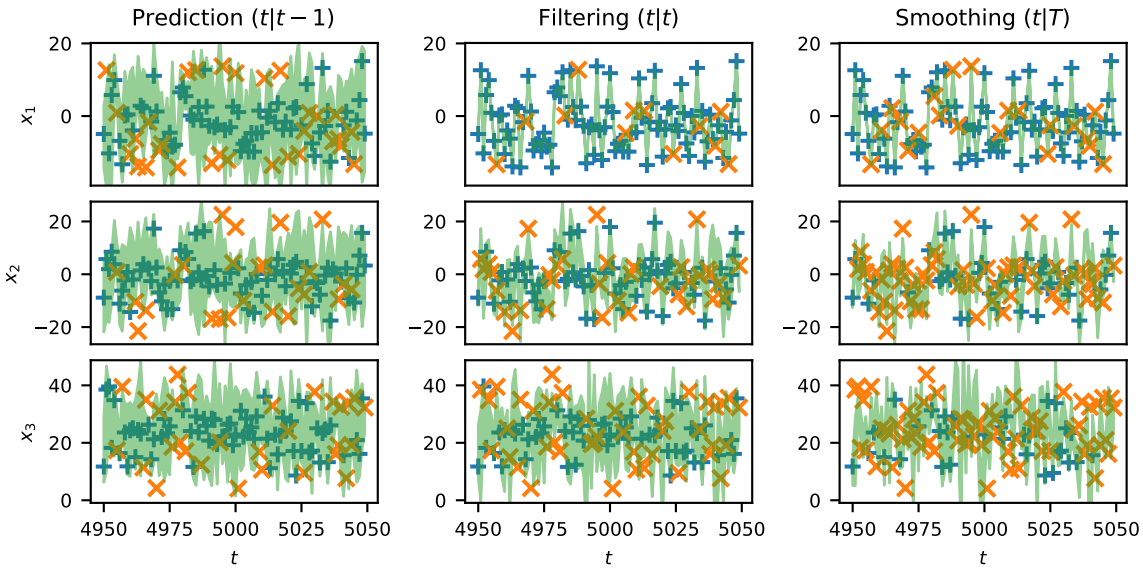

Figure 7: Trajectory excerpt for Kalman filter UNSCENTED'95 in the Lorenz system state estimation problem. Plus sign indicates hit; cross indicates miss: 10 crosses is best for nominal coverage.

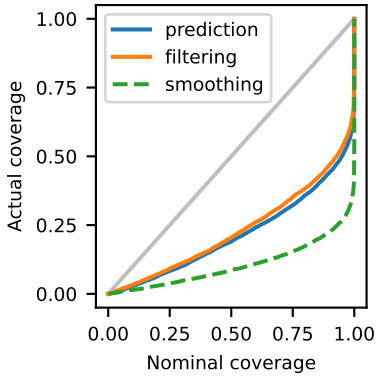

Figure 8: Coverage for Kalman filter UNSCENTED'95 in the Lorenz system state estimation problem. Closer to identity is best.

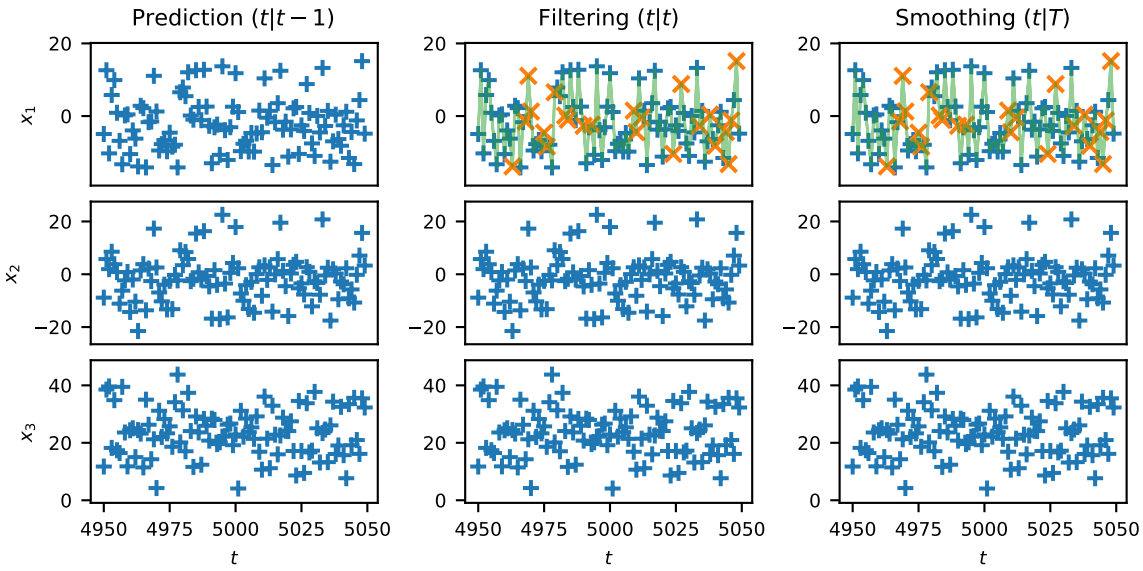

Figure 9: Trajectory excerpt for Kalman filter UNSCENTED'02 in the Lorenz system state estimation problem. Plus sign indicates hit; cross indicates miss: 10 crosses is best for nominal coverage.

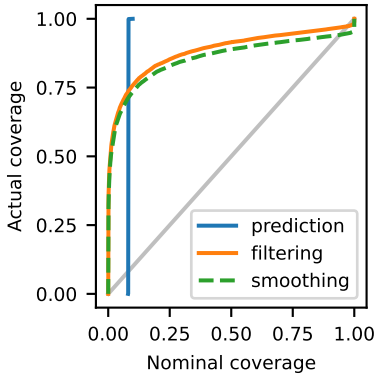

Figure 10: Coverage for Kalman filter UNSCENTED'02 in the Lorenz system state estimation problem. Closer to identity is best.

Appendix I. Full results for §7.1

Table 5: RMSE for different methods and tasks in the LTI state estimation problem.

Method	Task	Value ± Monte Carlo standard error
ANALYTIC	Prediction $(t\|t-1)$	$+1.377\,555 \times 10^{0} \pm 3.3 \times 10^{-2}$
	Filtering $(t\|t)$	$+1.310\,450 \times 10^{0} \pm 3.3 \times 10^{-2}$
	Smoothing $(t\|T)$	$+9.936\,777 \times 10^{-1} \pm 3.7 \times 10^{-2}$
ANALYTIC (RECAL)	Prediction $(t\|t-1)$	$+1.311\,325 \times 10^{0} \pm 8.5 \times 10^{-3}$
	Filtering $(t\|t)$	$+1.241\,469 \times 10^{0} \pm 8.5 \times 10^{-3}$
	Smoothing $(t\|T)$	$+8.996\,923 \times 10^{-1} \pm 5.3 \times 10^{-3}$
LINEAR	Prediction $(t\|t-1)$	$+3.145\,826 \times 10^{+1} \pm 8.0 \times 10^{-1}$
	Filtering $(t\|t)$	$+3.152\,544 \times 10^{+1} \pm 8.0 \times 10^{-1}$
	Smoothing $(t\|T)$	$+3.203\,451 \times 10^{+1} \pm 8.1 \times 10^{-1}$
LINEAR (RECAL)	Prediction $(t\|t-1)$	$+9.831\,827 \times 10^{0} \pm 7.0 \times 10^{-1}$
	Filtering $(t\|t)$	$+9.901\,705 \times 10^{0} \pm 7.0 \times 10^{-1}$
	Smoothing $(t\|T)$	$+1.085\,636 \times 10^{+1} \pm 7.3 \times 10^{-1}$
UNSCENTED'95	Prediction $(t\|t-1)$	$+6.560\,435 \times 10^{0} \pm 6.8 \times 10^{-1}$
	Filtering $(t\|t)$	$+6.541\,330 \times 10^{0} \pm 6.8 \times 10^{-1}$
	Smoothing $(t\|T)$	$+6.613\,449 \times 10^{0} \pm 6.8 \times 10^{-1}$
UNSCENTED'95 (RECAL)	Prediction $(t\|t-1)$	$+1.676\,370 \times 10^{0} \pm 5.5 \times 10^{-2}$
	Filtering $(t\|t)$	$+1.612\,820 \times 10^{0} \pm 5.7 \times 10^{-2}$
	Smoothing $(t\|T)$	$+1.316\,173 \times 10^{0} \pm 6.9 \times 10^{-2}$
UNSCENTED'02	Prediction $(t\|t-1)$	$+2.439\,038 \times 10^{+1} \pm 7.6 \times 10^{-1}$
	Filtering $(t\|t)$	$+2.457\,422 \times 10^{+1} \pm 7.7 \times 10^{-1}$
	Smoothing $(t\|T)$	$+2.558\,191 \times 10^{+1} \pm 8.1 \times 10^{-1}$
UNSCENTED'02 (RECAL)	Prediction $(t\|t-1)$	$+2.977\,201 \times 10^{0} \pm 1.2 \times 10^{-1}$
	Filtering $(t\|t)$	$+2.979\,570 \times 10^{0} \pm 1.3 \times 10^{-1}$
	Smoothing $(t\|T)$	$+2.847\,319 \times 10^{0} \pm 1.6 \times 10^{-1}$

Table 6: Coverage at 95% for different methods and tasks in the LTI state estimation problem.

Method	Task	Value $\pm$ Monte Carlo standard error
ANALYTIC	Prediction $(t\|t-1)$	$+9.432\,500 \times 10^{-1} \pm 1.7 \times 10^{-3}$
	Filtering $(t\|t)$	$+9.414\,889 \times 10^{-1} \pm 1.6 \times 10^{-3}$
	Smoothing $(t\|T)$	$+9.379\,111 \times 10^{-1} \pm 2.0 \times 10^{-3}$
ANALYTIC (RECAL)	Prediction $(t\|t-1)$	$+9.577\,944 \times 10^{-1} \pm 1.0 \times 10^{-3}$
	Filtering $(t\|t)$	$+9.601\,778 \times 10^{-1} \pm 8.8 \times 10^{-4}$
	Smoothing $(t\|T)$	$+9.634\,222 \times 10^{-1} \pm 7.3 \times 10^{-4}$
LINEAR	Prediction $(t\|t-1)$	$+2.842\,667 \times 10^{-1} \pm 7.1 \times 10^{-3}$
	Filtering $(t\|t)$	$+2.583\,111 \times 10^{-1} \pm 6.5 \times 10^{-3}$
	Smoothing $(t\|T)$	$+1.537\,611 \times 10^{-1} \pm 5.0 \times 10^{-3}$
LINEAR (RECAL)	Prediction $(t\|t-1)$	$+6.818\,222 \times 10^{-1} \pm 7.3 \times 10^{-3}$
	Filtering $(t\|t)$	$+6.641\,333 \times 10^{-1} \pm 7.3 \times 10^{-3}$
	Smoothing $(t\|T)$	$+5.557\,056 \times 10^{-1} \pm 8.1 \times 10^{-3}$
UNSCENTED'95	Prediction $(t\|t-1)$	$+8.226\,000 \times 10^{-1} \pm 6.1 \times 10^{-3}$
	Filtering $(t\|t)$	$+8.011\,222 \times 10^{-1} \pm 6.0 \times 10^{-3}$
	Smoothing $(t\|T)$	$+7.365\,500 \times 10^{-1} \pm 7.4 \times 10^{-3}$
UNSCENTED'95 (RECAL)	Prediction $(t\|t-1)$	$+9.334\,389 \times 10^{-1} \pm 2.2 \times 10^{-3}$
	Filtering $(t\|t)$	$+9.295\,889 \times 10^{-1} \pm 2.0 \times 10^{-3}$
	Smoothing $(t\|T)$	$+9.159\,167 \times 10^{-1} \pm 2.3 \times 10^{-3}$
UNSCENTED'02	Prediction $(t\|t-1)$	$+5.764\,778 \times 10^{-1} \pm 8.1 \times 10^{-3}$
	Filtering $(t\|t)$	$+5.592\,444 \times 10^{-1} \pm 7.9 \times 10^{-3}$
	Smoothing $(t\|T)$	$+4.547\,722 \times 10^{-1} \pm 9.2 \times 10^{-3}$
UNSCENTED'02 (RECAL)	Prediction $(t\|t-1)$	$+9.097\,889 \times 10^{-1} \pm 2.5 \times 10^{-3}$
	Filtering $(t\|t)$	$+9.071\,000 \times 10^{-1} \pm 2.6 \times 10^{-3}$
	Smoothing $(t\|T)$	$+8.839\,278 \times 10^{-1} \pm 3.9 \times 10^{-3}$

Table 7: Volume at 95% for different methods and tasks in the LTI state estimation problem.

Method	Task	Value $\pm$ Monte Carlo standard error
ANALYTIC	Prediction $(t\|t-1)$	$+1.616\,272 \times 10^{-1} \pm 2.7 \times 10^{-4}$
	Filtering $(t\|t)$	$+1.166\,921 \times 10^{-1} \pm 2.5 \times 10^{-4}$
	Smoothing $(t\|T)$	$+2.748\,990 \times 10^{-2} \pm 8.1 \times 10^{-5}$
ANALYTIC (RECAL)	Prediction $(t\|t-1)$	$+1.831\,996 \times 10^{-1} \pm 2.6 \times 10^{-4}$
	Filtering $(t\|t)$	$+1.363\,137 \times 10^{-1} \pm 2.3 \times 10^{-4}$
	Smoothing $(t\|T)$	$+3.361\,851 \times 10^{-2} \pm 7.5 \times 10^{-5}$
LINEAR	Prediction $(t\|t-1)$	$+1.233\,095 \times 10^{-1} \pm 4.8 \times 10^{-4}$
	Filtering $(t\|t)$	$+8.549\,655 \times 10^{-2} \pm 3.4 \times 10^{-4}$
	Smoothing $(t\|T)$	$+1.737\,160 \times 10^{-2} \pm 9.4 \times 10^{-5}$
LINEAR (RECAL)	Prediction $(t\|t-1)$	$+2.571\,232 \times 10^{-1} \pm 1.5 \times 10^{-3}$
	Filtering $(t\|t)$	$+2.026\,725 \times 10^{-1} \pm 1.4 \times 10^{-3}$
	Smoothing $(t\|T)$	$+5.929\,100 \times 10^{-2} \pm 7.7 \times 10^{-4}$
UNSCENTED'95	Prediction $(t\|t-1)$	$+1.389\,514 \times 10^{-1} \pm 3.1 \times 10^{-4}$
	Filtering $(t\|t)$	$+9.609\,799 \times 10^{-2} \pm 2.5 \times 10^{-4}$
	Smoothing $(t\|T)$	$+1.977\,639 \times 10^{-2} \pm 6.5 \times 10^{-5}$
UNSCENTED'95 (RECAL)	Prediction $(t\|t-1)$	$+1.925\,837 \times 10^{-1} \pm 4.7 \times 10^{-4}$
	Filtering $(t\|t)$	$+1.435\,821 \times 10^{-1} \pm 4.2 \times 10^{-4}$
	Smoothing $(t\|T)$	$+3.384\,204 \times 10^{-2} \pm 1.7 \times 10^{-4}$
UNSCENTED'02	Prediction $(t\|t-1)$	$+1.823\,639 \times 10^{-1} \pm 5.3 \times 10^{-4}$
	Filtering $(t\|t)$	$+1.371\,937 \times 10^{-1} \pm 4.7 \times 10^{-4}$
	Smoothing $(t\|T)$	$+3.178\,623 \times 10^{-2} \pm 2.1 \times 10^{-4}$
UNSCENTED'02 (RECAL)	Prediction $(t\|t-1)$	$+3.317\,351 \times 10^{-1} \pm 1.3 \times 10^{-3}$
	Filtering $(t\|t)$	$+2.798\,133 \times 10^{-1} \pm 1.4 \times 10^{-3}$
	Smoothing $(t\|T)$	$+9.692\,433 \times 10^{-2} \pm 9.6 \times 10^{-4}$

Table 8: Cross entropy for different methods and tasks in the LTI state estimation problem.

Method	Task	Value $\pm$ Monte Carlo standard error
ANALYTIC	Prediction $(t\|t-1)$	$-1.726\,823 \times 10^{0} \pm 7.9 \times 10^{-1}$
	Filtering $(t\|t)$	$-4.652\,060 \times 10^{-1} \pm 2.3 \times 10^{0}$
	Smoothing $(t\|T)$	$+1.083\,637 \times 10^{-1} \pm 4.0 \times 10^{0}$
ANALYTIC (RECAL)	Prediction $(t\|t-1)$	$-2.725\,533 \times 10^{0} \pm 9.6 \times 10^{-3}$
	Filtering $(t\|t)$	$-3.257\,829 \times 10^{0} \pm 8.5 \times 10^{-3}$
	Smoothing $(t\|T)$	$-4.640\,158 \times 10^{0} \pm 7.2 \times 10^{-3}$
LINEAR	Prediction $(t\|t-1)$	$+2.362\,394 \times 10^{+4} \pm 1.2 \times 10^{+3}$
	Filtering $(t\|t)$	$+1.794\,416 \times 10^{+5} \pm 1.1 \times 10^{+4}$
	Smoothing $(t\|T)$	$+2.116\,272 \times 10^{+5} \pm 1.2 \times 10^{+4}$
LINEAR (RECAL)	Prediction $(t\|t-1)$	$+1.027\,865 \times 10^{+2} \pm 1.1 \times 10^{+1}$
	Filtering $(t\|t)$	$+3.256\,018 \times 10^{+2} \pm 7.0 \times 10^{+1}$
	Smoothing $(t\|T)$	$+4.992\,393 \times 10^{+2} \pm 8.8 \times 10^{+1}$
UNSCENTED'95	Prediction $(t\|t-1)$	$+9.175\,953 \times 10^{+2} \pm 1.9 \times 10^{+2}$
	Filtering $(t\|t)$	$+3.885\,867 \times 10^{+3} \pm 8.5 \times 10^{+2}$
	Smoothing $(t\|T)$	$+5.232\,937 \times 10^{+3} \pm 1.1 \times 10^{+3}$
UNSCENTED'95 (RECAL)	Prediction $(t\|t-1)$	$-1.694\,058 \times 10^{0} \pm 2.1 \times 10^{-1}$
	Filtering $(t\|t)$	$-1.683\,487 \times 10^{0} \pm 3.6 \times 10^{-1}$
	Smoothing $(t\|T)$	$-2.069\,659 \times 10^{0} \pm 6.4 \times 10^{-1}$
UNSCENTED'02	Prediction $(t\|t-1)$	$+1.119\,020 \times 10^{+4} \pm 6.4 \times 10^{+2}$
	Filtering $(t\|t)$	$+6.514\,542 \times 10^{+4} \pm 4.4 \times 10^{+3}$
	Smoothing $(t\|T)$	$+8.183\,052 \times 10^{+4} \pm 5.2 \times 10^{+3}$
UNSCENTED'02 (RECAL)	Prediction $(t\|t-1)$	$+9.706\,521 \times 10^{-2} \pm 2.2 \times 10^{-1}$
	Filtering $(t\|t)$	$+1.388\,761 \times 10^{0} \pm 7.4 \times 10^{-1}$
	Smoothing $(t\|T)$	$+3.184\,046 \times 10^{0} \pm 1.0 \times 10^{0}$

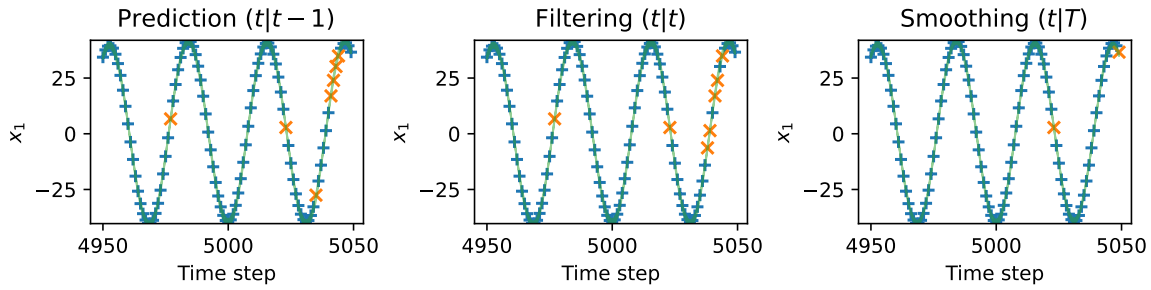

Figure 11: Trajectory excerpt for Kalman filter ANALYTIC in the LTI state estimation problem. Plus sign indicates hit; cross indicates miss. Expect 10 misses for nominal 90% coverage.

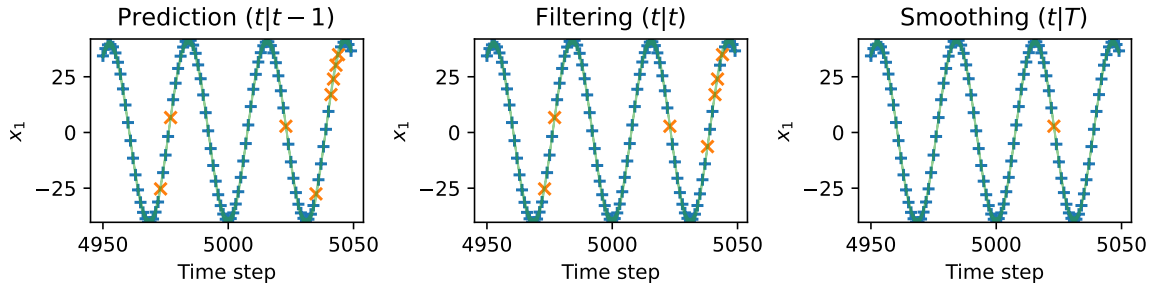

Figure 12: Trajectory excerpt for Kalman filter ANALYTIC (RECAL) in the LTI state estimation problem. Plus sign indicates hit; cross indicates miss. Expect 10 misses for nominal 90% coverage.

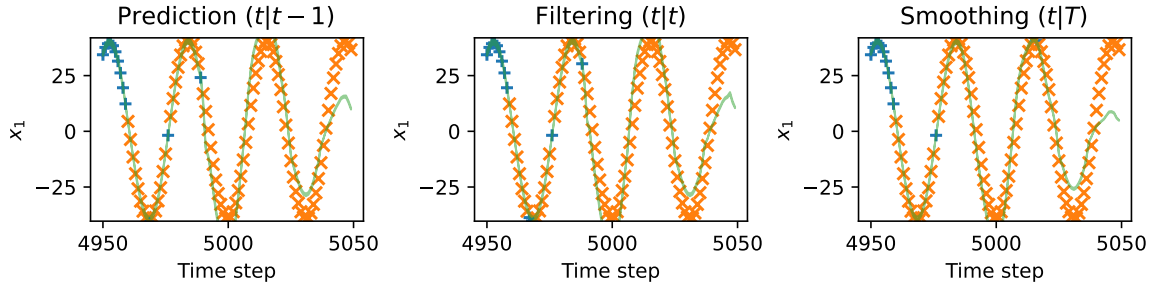

Figure 13: Trajectory excerpt for Kalman filter LINEAR in the LTI state estimation problem. Plus sign indicates hit; cross indicates miss. Expect 10 misses for nominal 90% coverage.

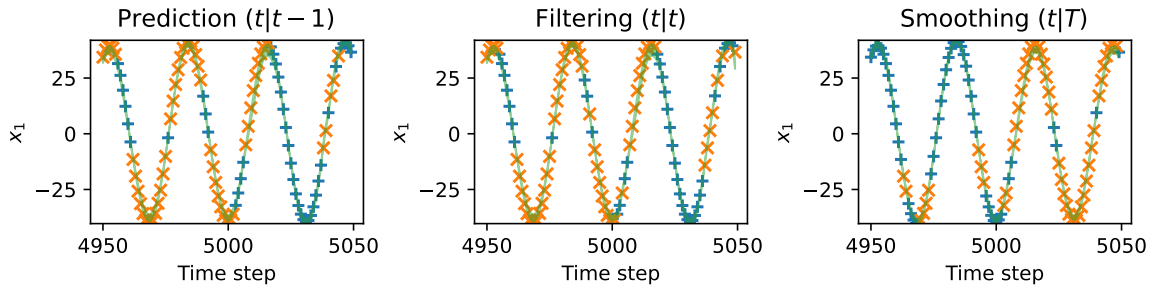

Figure 14: Trajectory excerpt for Kalman filter LINEAR (RECAL) in the LTI state estimation problem. Plus sign indicates hit; cross indicates miss. Expect 10 misses for nominal 90% coverage.

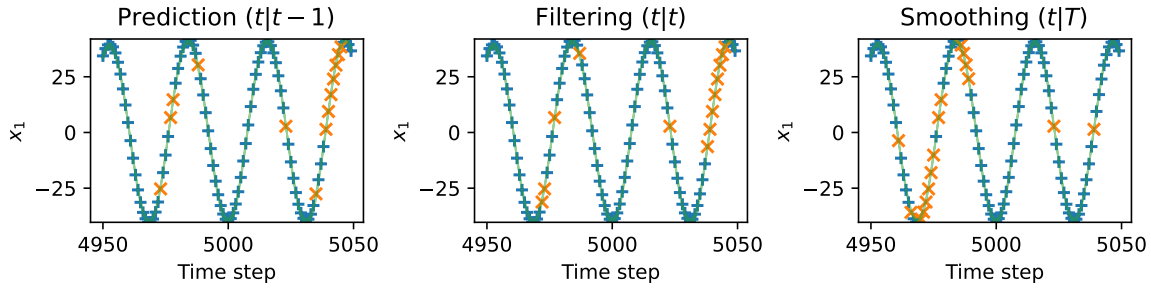

Figure 15: Trajectory excerpt for Kalman filter UNSCENTED'95 in the LTI state estimation problem. Plus sign indicates hit; cross indicates miss. Expect 10 misses for nominal 90% coverage.

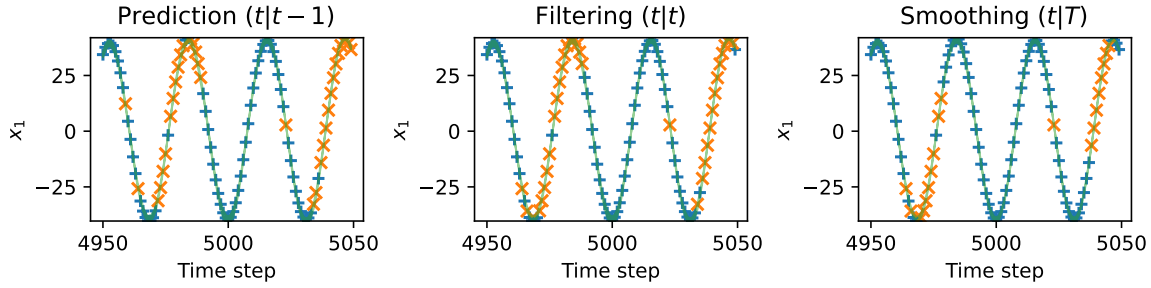

Figure 16: Trajectory excerpt for Kalman filter UNSCENTED'95 (RECAL) in the LTI state estimation problem. Plus sign indicates hit; cross indicates miss. Expect 10 misses for nominal 90% coverage.

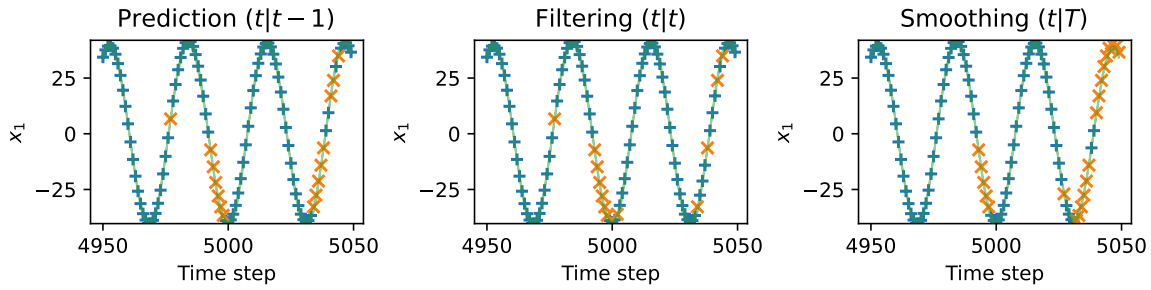

Figure 17: Trajectory excerpt for Kalman filter UNSCENTED'02 in the LTI state estimation problem. Plus sign indicates hit; cross indicates miss. Expect 10 misses for nominal 90% coverage.

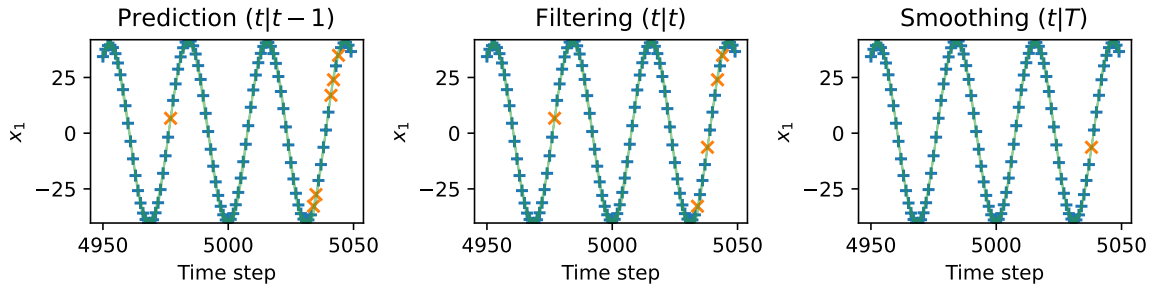

Figure 18: Trajectory excerpt for Kalman filter UNSCENTED'02 (RECAL) in the LTI state estimation problem. Plus sign indicates hit; cross indicates miss. Expect 10 misses for nominal 90% coverage.

I.1. Effect of recalibration

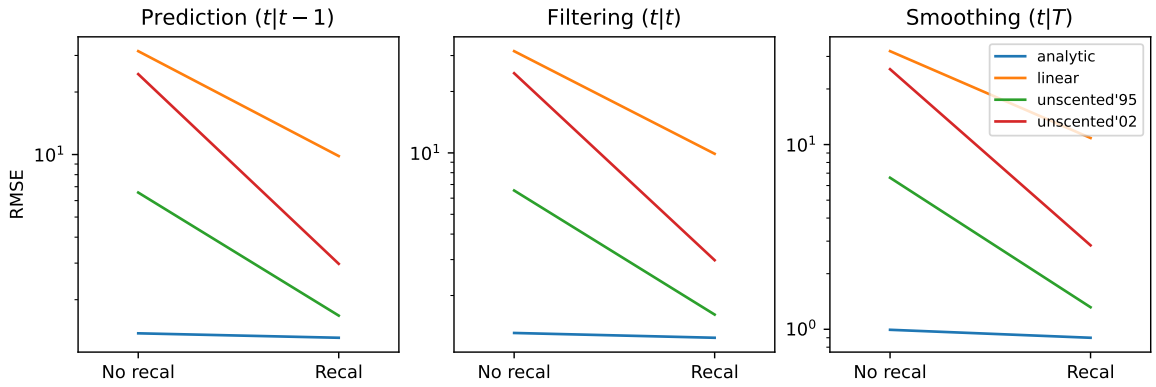

Figure 19: Effect of recalibration on RMSE for each method and inference task in the LTI state estimation problem. Each line segment connects a method's no-recal value to its recal value; vertical axis is log-scaled.

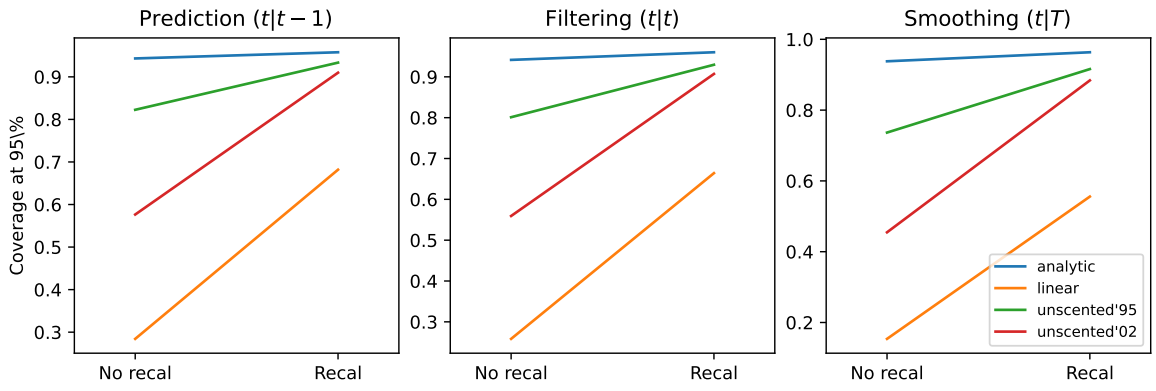

Figure 20: Effect of recalibration on 95% coverage.

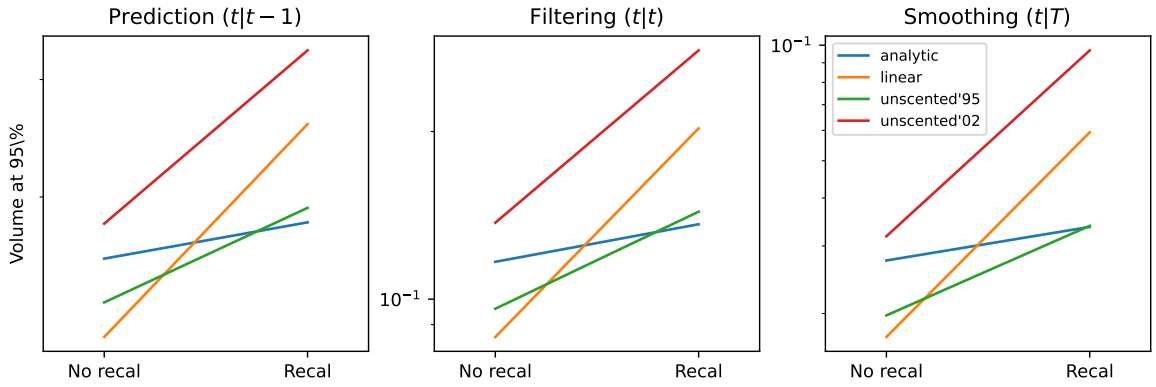

Figure 21: Effect of recalibration on 95% confidence volume (log-scaled).

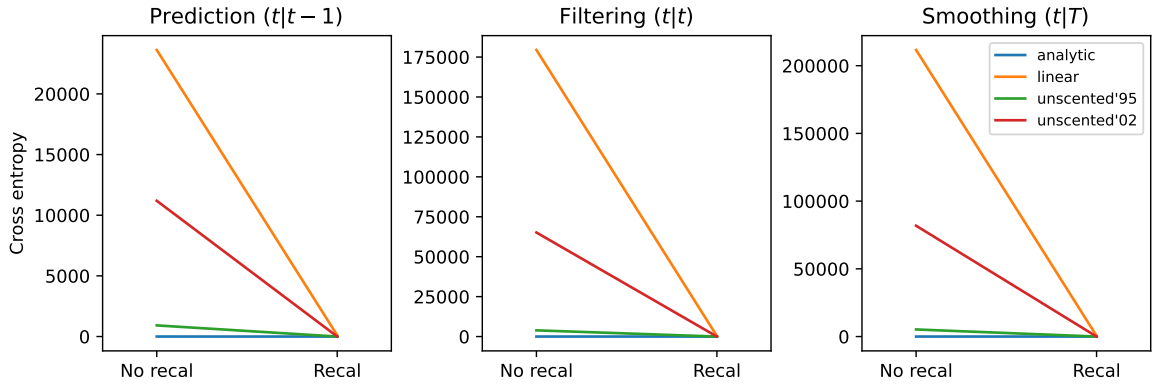

Figure 22: Effect of recalibration on cross entropy.

Appendix J. Full results for §7.2

Table 9: LQR performance of different methods and tasks in the LTI regulation problem.

Method	Task	Value ± Monte Carlo standard error
ANALYTIC	State cost / LQR	$+1.053\,475 \times 10^{0} \pm 3.2 \times 10^{-3}$
	Control cost / LQR	$+9.999\,039 \times 10^{-1} \pm 3.0 \times 10^{-3}$
	Total cost / LQR	$+1.046\,893 \times 10^{0} \pm 3.2 \times 10^{-3}$
ANALYTIC (RECAL)	State cost / LQR	$+1.049\,540 \times 10^{0} \pm 3.4 \times 10^{-3}$
	Control cost / LQR	$+1.027\,164 \times 10^{0} \pm 3.2 \times 10^{-3}$
	Total cost / LQR	$+1.046\,791 \times 10^{0} \pm 3.3 \times 10^{-3}$
LINEAR	State cost / LQR	$+1.914\,271 \times 10^{+6} \pm 6.8 \times 10^{+5}$
	Control cost / LQR	$+1.687\,060 \times 10^{+2} \pm 4.3 \times 10^{0}$
	Total cost / LQR	$+1.679\,098 \times 10^{+6} \pm 5.9 \times 10^{+5}$
LINEAR (RECAL)	State cost / LQR	$+5.340\,074 \times 10^{+6} \pm 4.7 \times 10^{+5}$
	Control cost / LQR	$+3.149\,745 \times 10^{0} \pm 1.8 \times 10^{-1}$
	Total cost / LQR	$+4.683\,974 \times 10^{+6} \pm 4.2 \times 10^{+5}$
UNSCENTED'95	State cost / LQR	$+2.838\,426 \times 10^{+5} \pm 1.4 \times 10^{+5}$
	Control cost / LQR	$+1.689\,836 \times 10^{+2} \pm 1.1 \times 10^{+1}$
	Total cost / LQR	$+2.489\,894 \times 10^{+5} \pm 1.3 \times 10^{+5}$
UNSCENTED'95 (RECAL)	State cost / LQR	$+9.005\,846 \times 10^{0} \pm 2.2 \times 10^{0}$
	Control cost / LQR	$+3.704\,188 \times 10^{0} \pm 1.9 \times 10^{-1}$
	Total cost / LQR	$+8.354\,465 \times 10^{0} \pm 2.0 \times 10^{0}$
UNSCENTED'02	State cost / LQR	$+5.775\,930 \times 10^{+5} \pm 2.6 \times 10^{+5}$
	Control cost / LQR	$+3.378\,205 \times 10^{+2} \pm 1.2 \times 10^{+1}$
	Total cost / LQR	$+5.066\,693 \times 10^{+5} \pm 2.3 \times 10^{+5}$
UNSCENTED'02 (RECAL)	State cost / LQR	$+1.693\,187 \times 10^{+3} \pm 3.0 \times 10^{+2}$
	Control cost / LQR	$+7.595\,572 \times 10^{-3} \pm 4.4 \times 10^{-3}$
	Total cost / LQR	$+1.485\,157 \times 10^{+3} \pm 2.7 \times 10^{+2}$

Table 10: RMSE of the Kalman filters in closed loop in the LTI regulation problem.

Method	Task	Value $\pm$ Monte Carlo standard error
ANALYTIC	Filtering	$+3.568\,368 \times 10^{-2} \pm 8.3 \times 10^{-5}$
ANALYTIC (RECAL)	Filtering	$+3.590\,258 \times 10^{-2} \pm 7.6 \times 10^{-5}$
LINEAR	Filtering	$+1.780\,201 \times 10^{+2} \pm 2.9 \times 10^{+1}$
LINEAR (RECAL)	Filtering	$+3.583\,328 \times 10^{+2} \pm 2.1 \times 10^{+1}$
UNSCENTED'95	Filtering	$+5.917\,882 \times 10^{+1} \pm 1.4 \times 10^{+1}$
UNSCENTED'95 (RECAL)	Filtering	$+4.307\,320 \times 10^{-1} \pm 4.0 \times 10^{-2}$
UNSCENTED'02	Filtering	$+8.869\,023 \times 10^{+1} \pm 1.9 \times 10^{+1}$
UNSCENTED'02 (RECAL)	Filtering	$+6.091\,593 \times 10^{0} \pm 5.6 \times 10^{-1}$

Table 11: Coverage at 95% of the Kalman filters in closed loop in the LTI regulation problem.

Method	Task	Value $\pm$ Monte Carlo standard error
ANALYTIC	Filtering	$+9.123\,362 \times 10^{-1} \pm 4.7 \times 10^{-4}$
ANALYTIC (RECAL)	Filtering	$+9.966\,397 \times 10^{-1} \pm 1.4 \times 10^{-4}$
LINEAR	Filtering	$+2.035\,204 \times 10^{-3} \pm 2.9 \times 10^{-4}$
LINEAR (RECAL)	Filtering	$+4.074\,407 \times 10^{-2} \pm 9.2 \times 10^{-3}$
UNSCENTED'95	Filtering	$+1.837\,184 \times 10^{-2} \pm 1.5 \times 10^{-3}$
UNSCENTED'95 (RECAL)	Filtering	$+8.528\,403 \times 10^{-1} \pm 2.3 \times 10^{-3}$
UNSCENTED'02	Filtering	$+6.340\,634 \times 10^{-3} \pm 9.3 \times 10^{-4}$
UNSCENTED'02 (RECAL)	Filtering	$+9.464\,046 \times 10^{-1} \pm 9.7 \times 10^{-3}$

Table 12: Volume at 95% of the Kalman filters in closed loop in the LTI regulation problem.

Method	Task	Value $\pm$ Monte Carlo standard error
ANALYTIC	Filtering	$+4.513\,682 \times 10^{-4} \pm 4.6 \times 10^{-7}$
ANALYTIC (RECAL)	Filtering	$+2.381\,946 \times 10^{-3} \pm 8.2 \times 10^{-6}$
LINEAR	Filtering	$+2.647\,376 \times 10^{-3} \pm 7.9 \times 10^{-5}$
LINEAR (RECAL)	Filtering	— $\pm$ —
UNSCENTED'95	Filtering	$+3.470\,272 \times 10^{-3} \pm 2.6 \times 10^{-4}$
UNSCENTED'95 (RECAL)	Filtering	$+5.801\,317 \times 10^{-1} \pm 8.4 \times 10^{-3}$
UNSCENTED'02	Filtering	$+1.100\,528 \times 10^{-2} \pm 4.3 \times 10^{-4}$
UNSCENTED'02 (RECAL)	Filtering	$+6.157\,532 \times 10^{+2} \pm 6.5 \times 10^{0}$

Table 13: Cross entropy of the Kalman filters in closed loop in the LTI regulation problem.

Method	Task	Value $\pm$ Monte Carlo standard error
ANALYTIC	Filtering	$-8.217\,721 \times 10^{0} \pm 8.6 \times 10^{-3}$
ANALYTIC (RECAL)	Filtering	$-7.989\,011 \times 10^{0} \pm 2.5 \times 10^{-3}$
LINEAR	Filtering	$+2.249\,131 \times 10^{+9} \pm 8.5 \times 10^{+8}$
LINEAR (RECAL)	Filtering	— $\pm$ —
UNSCENTED'95	Filtering	$+3.491\,803 \times 10^{+7} \pm 1.7 \times 10^{+7}$
UNSCENTED'95 (RECAL)	Filtering	$+9.613\,007 \times 10^{-1} \pm 1.0 \times 10^{0}$
UNSCENTED'02	Filtering	$+1.806\,193 \times 10^{+8} \pm 7.9 \times 10^{+7}$
UNSCENTED'02 (RECAL)	Filtering	$+2.727\,023 \times 10^{+1} \pm 2.1 \times 10^{+1}$

Appendix K. Runtime

Tables 14 and 15 report per-call wall-clock time and XLA cost-analysis metrics for the Kalman `Predict`/`Update` and RTS `Predict`/`Update` primitives of Algorithms 1–2, evaluated in the Lorenz (§6) and LTI (§7.1) estimation problems. The Kalman and RTS `Update` rows for which recalibration has no effect are reported once; vanilla Kalman `Update` is likewise method-agnostic since it reduces to conditioning on a jointly Gaussian distribution. Benchmark inputs are drawn from random (non-data) distributions of the correct shape and dtype, since only the compiled graph governs runtime. Wall clock is reported as mean $\pm$ standard error over 10 repetitions on CPU (`jax_platform_name=cpu`, `jax_enable_x64=True`), each repetition sized so that the JIT-compiled call is invoked enough times to run for at least 0.2 s. FLOPs, bytes accessed, and transcendental-op counts come from `jax.jit(fn).lower().compile().cost_analysis()`.

All in all, the ANALYTIC operations are up to 10 times slower than conventional methods such as UNSCENTED'95, albeit faster than the FLOPs or other low-level operation counts would suggest. Though the ANALYTIC filter and smoother represent a performance-accuracy tradeoff, we reiterate from Kuang and Lin (2026) that they ultimately reside in the same complexity class as LINEAR (EKF).

Table 14: Per-call runtime of Kalman and RTS primitives in the Lorenz state estimation problem. Wall clock reports mean $\pm$ standard error over 10 repetitions on CPU.

Operation	Wall clock (μs)	FLOPs	Bytes accessed	Transcendentals
Kalman `Predict`				
ANALYTIC	2240.36 $\pm$ 39.46	11 046 598	3 254 294	103 680
LINEAR	258.85 $\pm$ 8.13	270 093	625 310	640
UNSCENTED'95	367.83 $\pm$ 13.85	472 859	398 116	2 240
UNSCENTED'02	290.36 $\pm$ 12.34	472 859	398 116	2 240
MEAN FIELD	1194.42 $\pm$ 43.57	11 087 679	3 262 598	103 680
Kalman `Update` *(vanilla)*				
	19.39 $\pm$ 0.57	24	449	0
RTS `Predict`				
ANALYTIC	2233.02 $\pm$ 22.62	12 676 065	3 567 919	113 565
LINEAR	102.22 $\pm$ 2.02	519 798	748 751	670
UNSCENTED'95	111.37 $\pm$ 4.31	520 803	429 842	2 345
UNSCENTED'02	87.90 $\pm$ 1.44	520 803	429 842	2 345
MEAN FIELD	720.53 $\pm$ 19.78	12 721 100	3 576 591	113 565
RTS `Update`				
	26.81 $\pm$ 1.08	175	1 889	0

Table 15: Per-call runtime of Kalman and RTS primitives in the LTI state-estimation problem. Wall clock reports mean ± standard error over 10 repetitions on CPU.

Operation	Wall clock (μs)	FLOPs	Bytes accessed	Transcendentals
Kalman `Predict`				
ANALYTIC	384.50 ± 6.13	697 938	1 438 494	63 000
LINEAR	47.78 ± 2.29	44 007	66 046	112
UNSCENTED'95	120.99 ± 3.60	112 632	78 692	1 456
UNSCENTED'02	112.43 ± 1.84	112 632	78 692	1 456
Kalman `Update` *(vanilla)*				
	20.38 ± 0.32	238	2 705	0
Kalman `Update` *(recalibrated)*				
ANALYTIC	398.84 ± 9.86	691 056	1 430 408	62 888
LINEAR	42.93 ± 1.77	36 796	57 144	56
UNSCENTED'95	90.63 ± 2.87	109 929	73 071	1 456
UNSCENTED'02	93.81 ± 5.76	109 929	73 071	1 456
RTS `Predict`				
ANALYTIC	29.60 ± 0.63	8 371	16 759	0
LINEAR	36.44 ± 0.57	10 197	19 927	0
UNSCENTED'95	52.87 ± 0.65	14 454	23 366	0
UNSCENTED'02	53.66 ± 1.32	14 454	23 366	0
RTS `Update`				
	26.06 ± 0.60	683	4 961	0