ASEN 5044, Fall 2024

Statistical Estimation for Dynamical Systems

Lecture 26:
The Kalman Filter (KF)

Prof. Nisar Ahmed (Nisar.Ahmed@Colorado.edu)

Tuesday 11/05/2024

University of Colorado **Boulder**

Announcements

- **HW 7 due this Thursday 11/07 on Gradescope**
- **Midterm 2: will be released Thursday 11/07, due Thurs 11/14 (1 week!)**
  - Will focus on HWs 5-7, Quizzes up to 7
  - **No office hours next week (can ask instructor + TFs for clarification on Midterm only)**
- **Final project descriptions + partner/group sign-ups – coming soon…**

Last Time…

- Flavors of dynamic state estimation (pure prediction, filtering, smoothing)
- Estimation for dynamic states (Linear Systems)
  - **Dynamic propagation of uncertain states** driven by AWGN (process noise)

    ("pure statistical state prediction": how does x estimate change if no y_k data, but only process noise w_k?)

want $\hat{x}_k^- = \text{argmin}\, tr(P_k^-) = \text{argmin}\, E(\underbrace{[x_k - \hat{x}_k^-]}_{e_k^-}[\cdots]^T[\cdots])$

$\hat{x}_k^- = E[x_k \mid \text{info from past}] \rightarrow \boxed{\hat{x}_k^- = F\hat{x}_{k-1}^- + Gu_{k-1}}$

$P_k^- = cov[x_k \mid \text{info from past}] \rightarrow \boxed{P_k^- = FP_{k-1}^-F^T + Q}$

} recursive

Today...

- **The Kalman Filter (KF)**: dynamic predictor-corrector state estimator

  ("dynamic RLLS": combine state prediction with RLLS updates at each time k, i.e. handle dynamics + process noise w_k + noisy measurements y_k)

  - Algorithm
  - Example
  - Important/useful properties

READ SIMON BOOK, CHAPTERS 5.5, 6.1-6.2

The Kalman Filter (KF): "The Whole Enchilada…"

- We are now fully equipped to solve state filtering problems and derive the KF
- Set up: same dynamics as for pure prediction, **but now with noisy y_k measurements**

$$x(k+1) = Fx(k) + Gu(k) + w(k), \qquad w(k) = \mathcal{N}(0, Q) \text{ (AWGN)}$$
$$y(k) = H(k)x(k) + v(k), \qquad v(k) = \mathcal{N}(0, R) \text{ (AWGN)}$$

H(k) (generally – other F, G, Q, R matrices can also vary w/ time!)

- First let's define the following state estimation error vector at each time step k:

$$e_k^+ = x_k - \hat{x}_k^+ \longrightarrow \hat{x}_k^+ = \hat{x}^+(k|k) = \text{"Filtered" state estimate given all data } y(1) \to y(k) \text{ (up to & including time k)}$$

- To get optimal recursive (dynamic) state estimator, define following cost function for each time k:

$$J(k) = E[e_k^{+T} e_k^+] = E(\text{tr}[e_k^+ e_k^{+T}]) = \text{tr}(E[e_k^+ e_k^{+T}]) = \text{trace}(P_k^+) = \text{sum of variances for state est. errors @ time k given } y(1) \to y(k)$$

Optimal filtering problem: suppose we have initial guess $\hat{x}_0^+$ & P_0^+ & $y(1), \ldots, y(k)$

→ How to get: $(\hat{x}_1^+, P_1^+)$, $(\hat{x}_2^+, P_2^+)$, …., $(\hat{x}_k^+, P_k^+)$, …

in order to minimize: $J(1)$, $J(2)$, …, $J(k)$, …

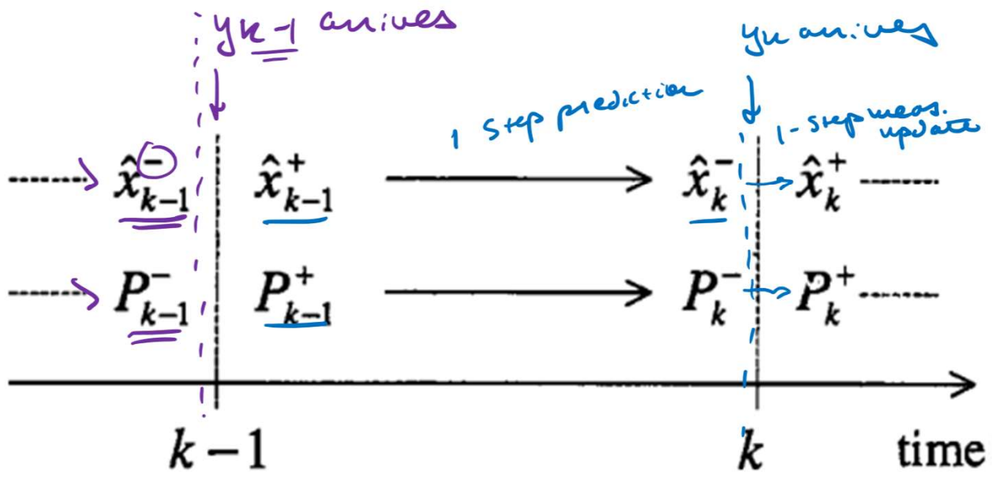
Kalman Filter (KF): Dynamic State "Predictor-Corrector" Recursion

- It turns out that the answer is <u>almost exactly same</u> as RLLS recursion, <u>but with one important addition to account for stochastic (linear) dynamics/time dependencies</u>
- KF uses two main stages to process measurements and estimates each time k=1,2,3,…
  - "**Time update step for time k+1**" (i.e. "**dynamics prediction/propagation step**"):

$$\hat{x}_{k+1}^{-} = F\hat{x}_{k}^{+} + Gu_k$$

$$P_{k+1}^{-} = FP_{k}^{+}F^T + Q$$

$$K_{k+1} = P_{k+1}^{-}H_{k+1}^T[H_{k+1}P_{k+1}^{-}H_{k+1}^T + R]^{-1}$$



- "**Measurement update step at time k+1**" ("**correction**"):

$$\hat{x}_{k+1}^{+} = \hat{x}_{k+1}^{-} + K_{k+1}(y_{k+1} - H_{k+1}\hat{x}_{k+1}^{-})$$

$$P_{k+1}^{+} = (I - K_{k+1}H_{k+1})P_{k+1}^{-}$$

KF Block Diagram: Full "Predictor-Corrector" Cycle

Assumed "true model" of system…

$$x(k+1) = Fx(k) + Gu(k) + w(k), \qquad w(k) = \mathcal{N}(0, Q) \text{ (AWGN)}$$
$$y(k+1) = H(k+1)x(k+1) + v(k+1), \qquad v(k+1) = \mathcal{N}(0, R) \text{ (AWGN)}$$

Initial state estimate $\hat{x}^+(0), P^+(0)$

u_k control input

1. KF Time Update ("Prediction Step") for time step k+1

$$\hat{x}^-_{k+1} = F\hat{x}^+_k + Gu_k$$
$$P^-_{k+1} = FP^+_k F^T + Q$$
$$K_{k+1} = P^-_{k+1}H^T_{k+1}[H_{k+1}P^-_{k+1}H^T_{k+1} + R]^{-1}$$

(cycle to next time step)

2. KF Measurement Update ("Correction Step") at time step k+1

$$\hat{x}^+_{k+1} = \hat{x}^-_{k+1} + K_{k+1}(y_{k+1} - H_{k+1}\hat{x}^-_{k+1})$$
$$P^+_{k+1} = (I - K_{k+1}H_{k+1})P^-_{k+1}$$

sensors y_{k+1}

state estimate $\hat{x}^+_{k+1}, P^+_{k+1}$

Example: 1D Robot: Part Deaux: Où suis-je maintenant??

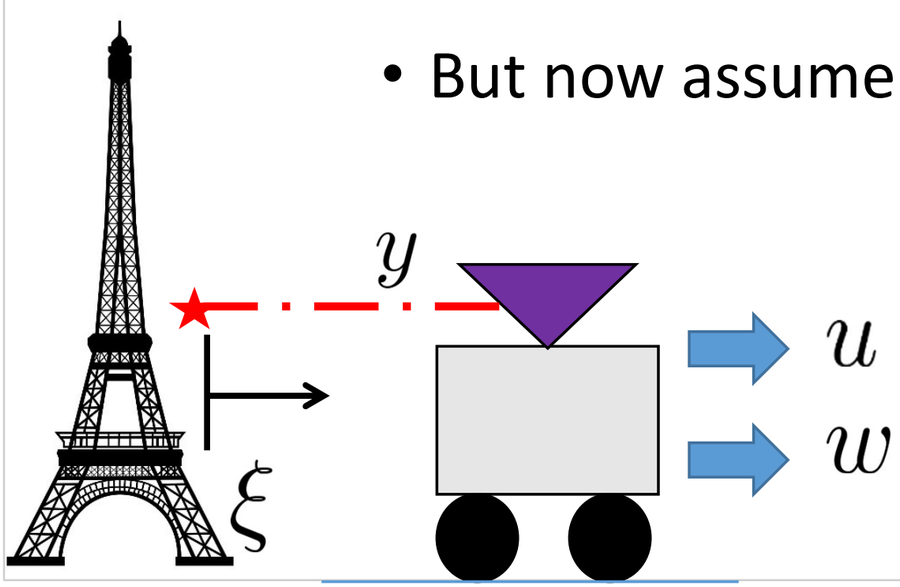
- Same DT model as before:

$$x(k) = [\xi(k), \dot{\xi(k)}]^T$$
$$u(k) = 2\cos(0.75t_k) \text{ (ZOH)}$$
$$w(k) \sim \mathcal{N}(0, Q) \text{ (AWGN)}$$
$$x(0) \sim \mathcal{N}(\mu_0, P_0), \text{ where } \mu_0 = [0,0]^T,\ P_0 = I_{2\times 2}$$

$$x(k+1) = Fx(k) + Gu(k) + w(k)$$

$$F = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \quad G = \begin{bmatrix} 0.5\Delta t^2 \\ \Delta t \end{bmatrix}$$

$$\Delta t = 0.1 \text{ sec}$$



- But now assume lidar measurements to surveyed landmark at origin:

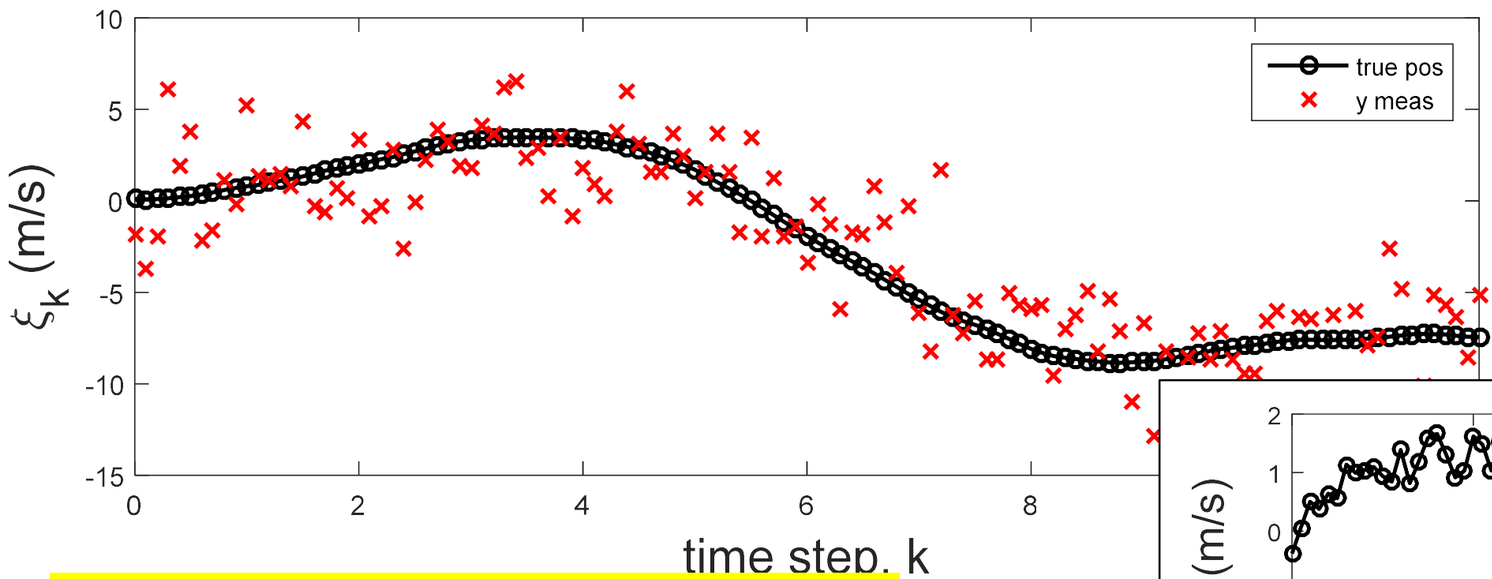
$$y(k) = Hx(k) + v(k)$$

$$H = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

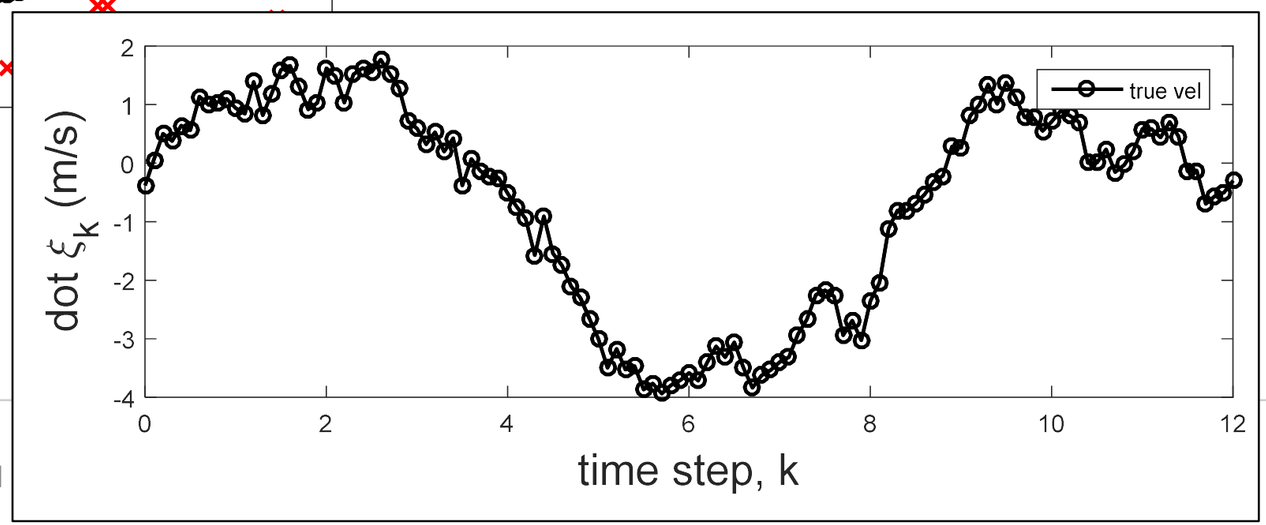
$$v(k) \sim \mathcal{N}(0, R) \text{ (AWGN)}$$

Monte Carlo True State and Measurement Simulation

- First, get a possible "ground truth" state history: sample p(x(0))~N(μ_0,P_0), then push through DT dynamics with sampled w(k)~N(0,Q) noise at each time k
- Then generate sensor measurements y(k) = Hx(k) + sampled noise v(k) ~N(0,R)
- Example for CT $\tilde{W}$ = 1 (m/s^2)^2 (use Van Loan's method to get DT Q), R = 0.5 m^2



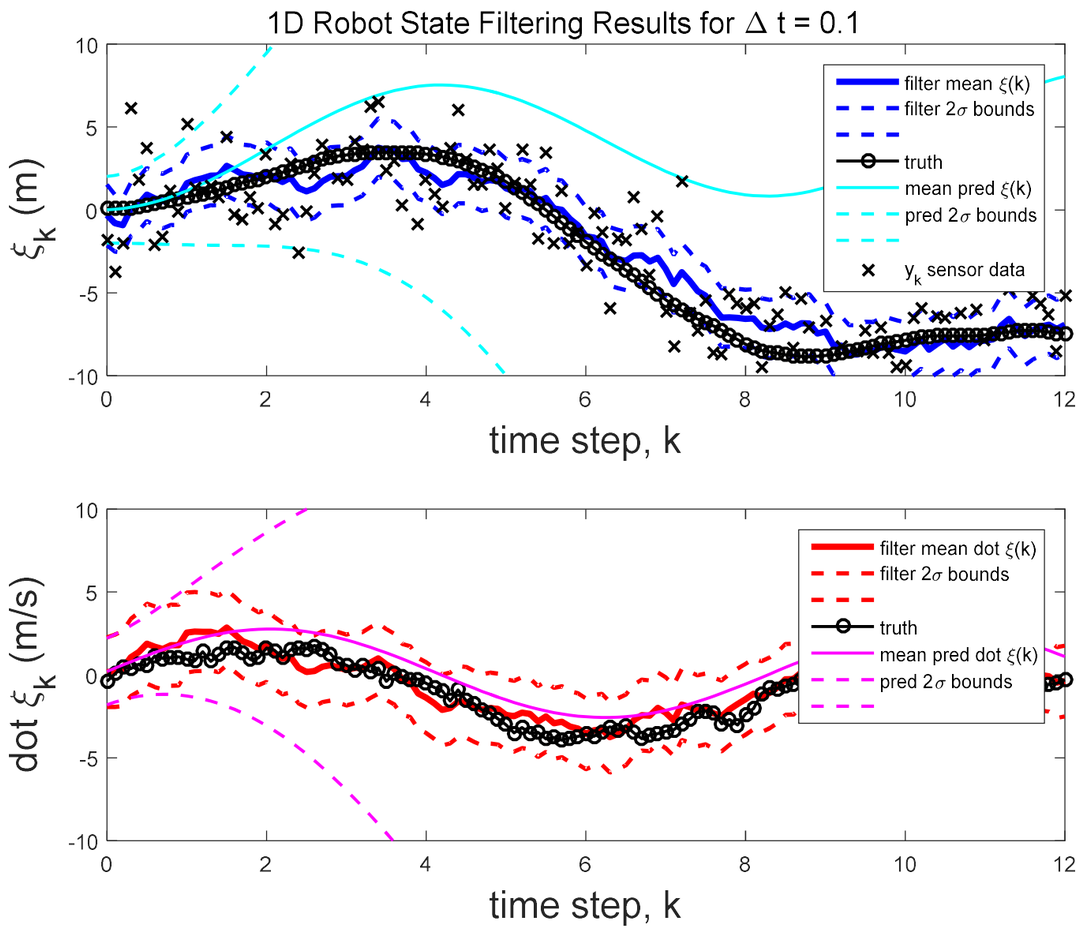
Then, apply KF algorithm from previous slides to estimate x(k) states at each time k using y(k) data only

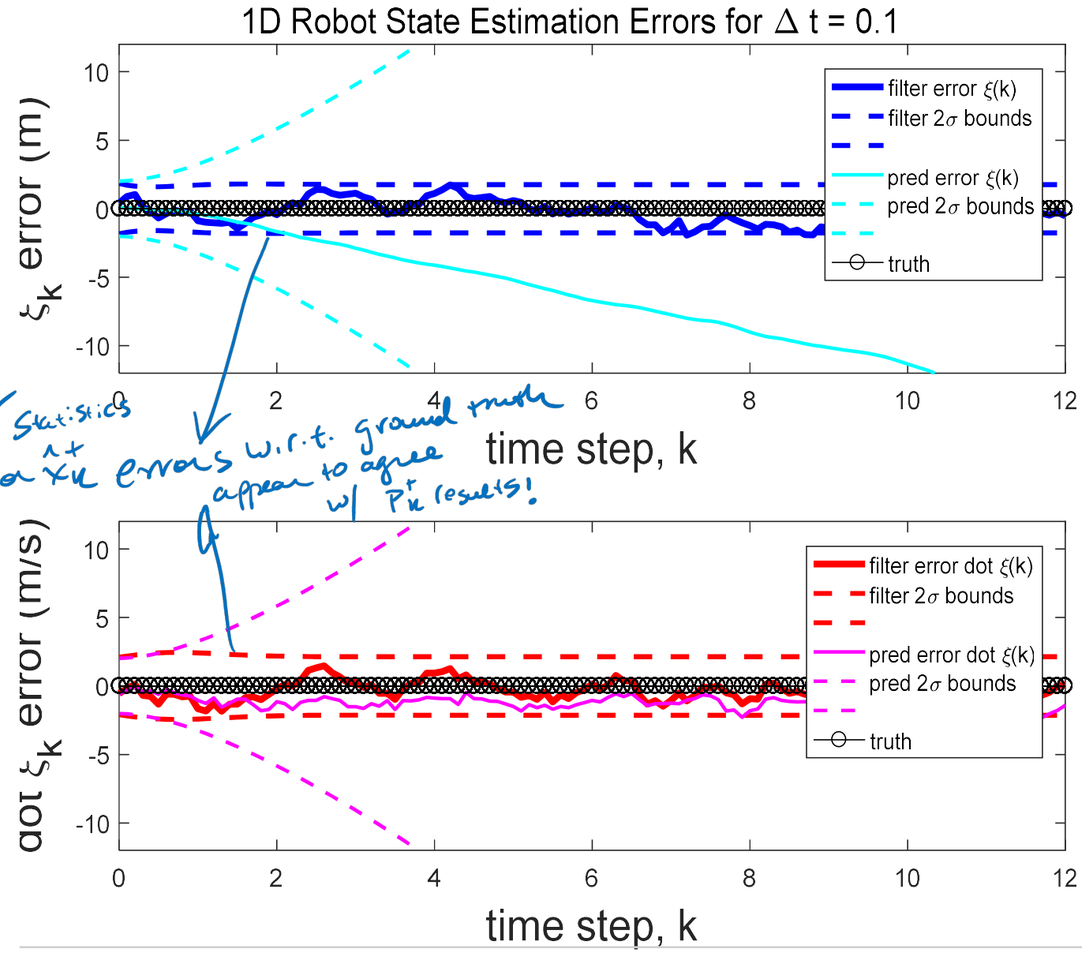


(note: the KF itself DOES NOT simulate process/measurement noise in its recursions!!! It accounts for these via Q and R!!!)

1D Robot Car with Lidar -- KF Results…

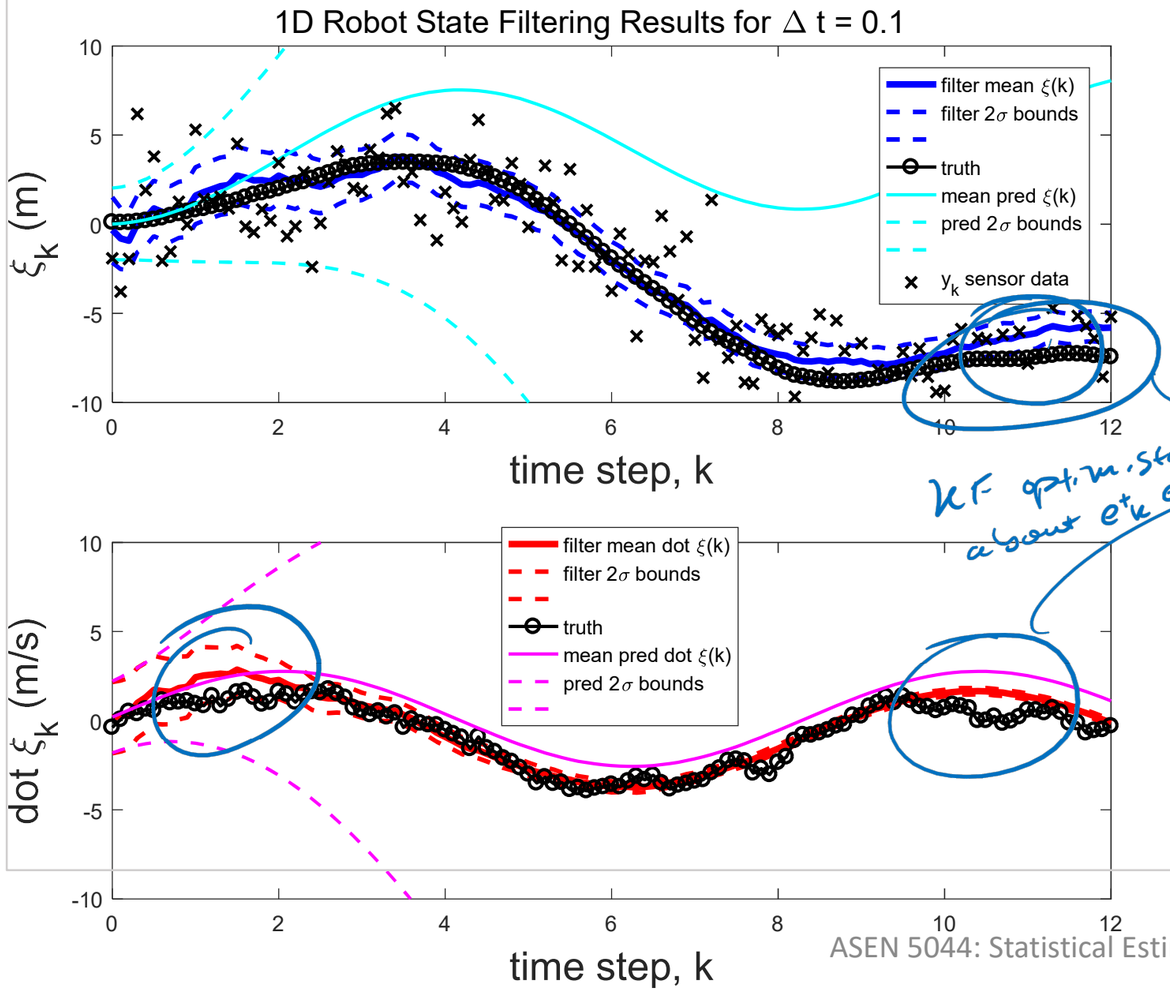
- KF keeps pretty good track of true states in this case (compare 2σ bounds vs. pure prediction)
- State estimation error e_k appears to be ~ 0 mean, and within KF's 2σ bounds according to covar P

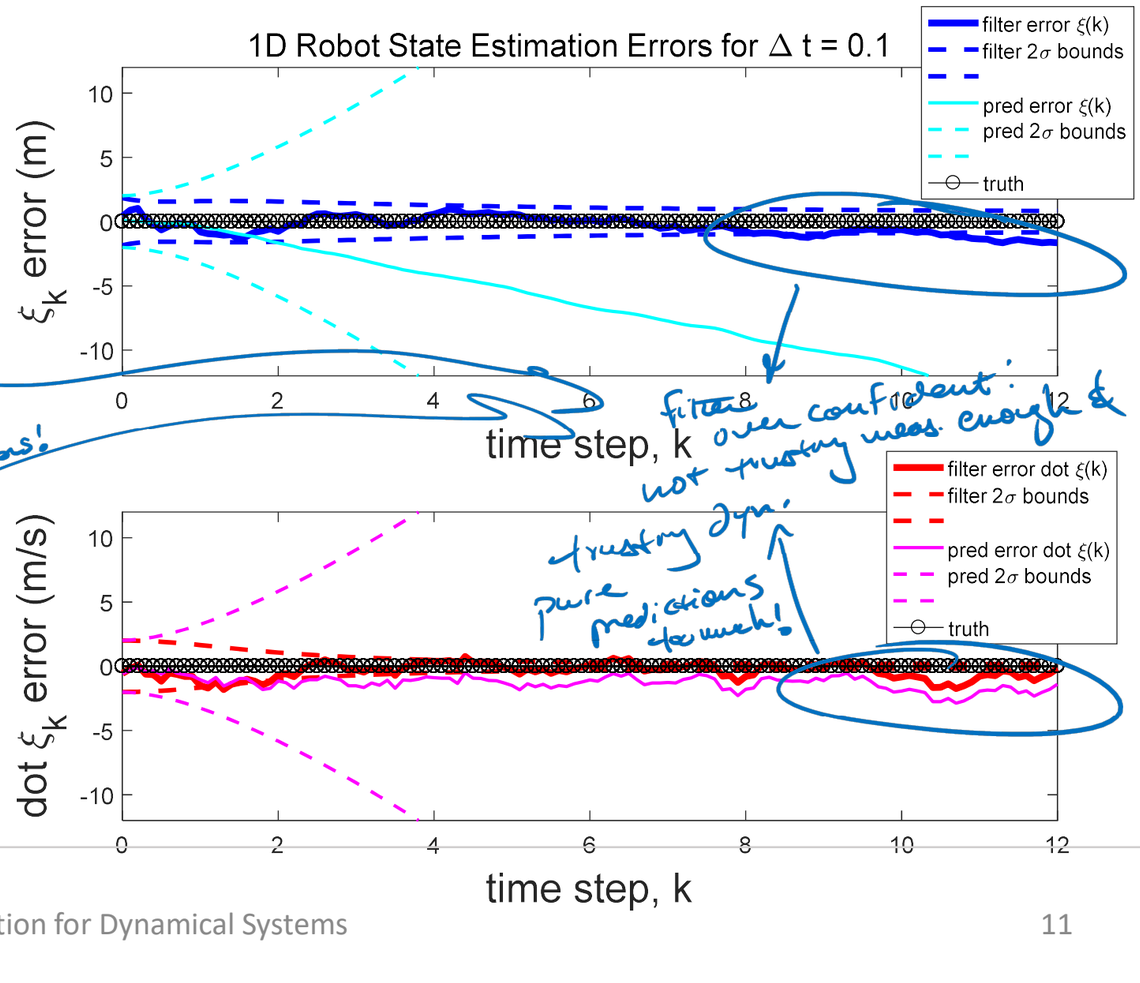




KF Results…

- See Matlab code: Lec26_1Drobotstatefilter.m
- What happens if our KF uses the incorrect values for the process noise covar Q?
- Example: say Q_{KF} = 0.001* Q (where Q_{KF} is what the KF "thinks" the real Q is)

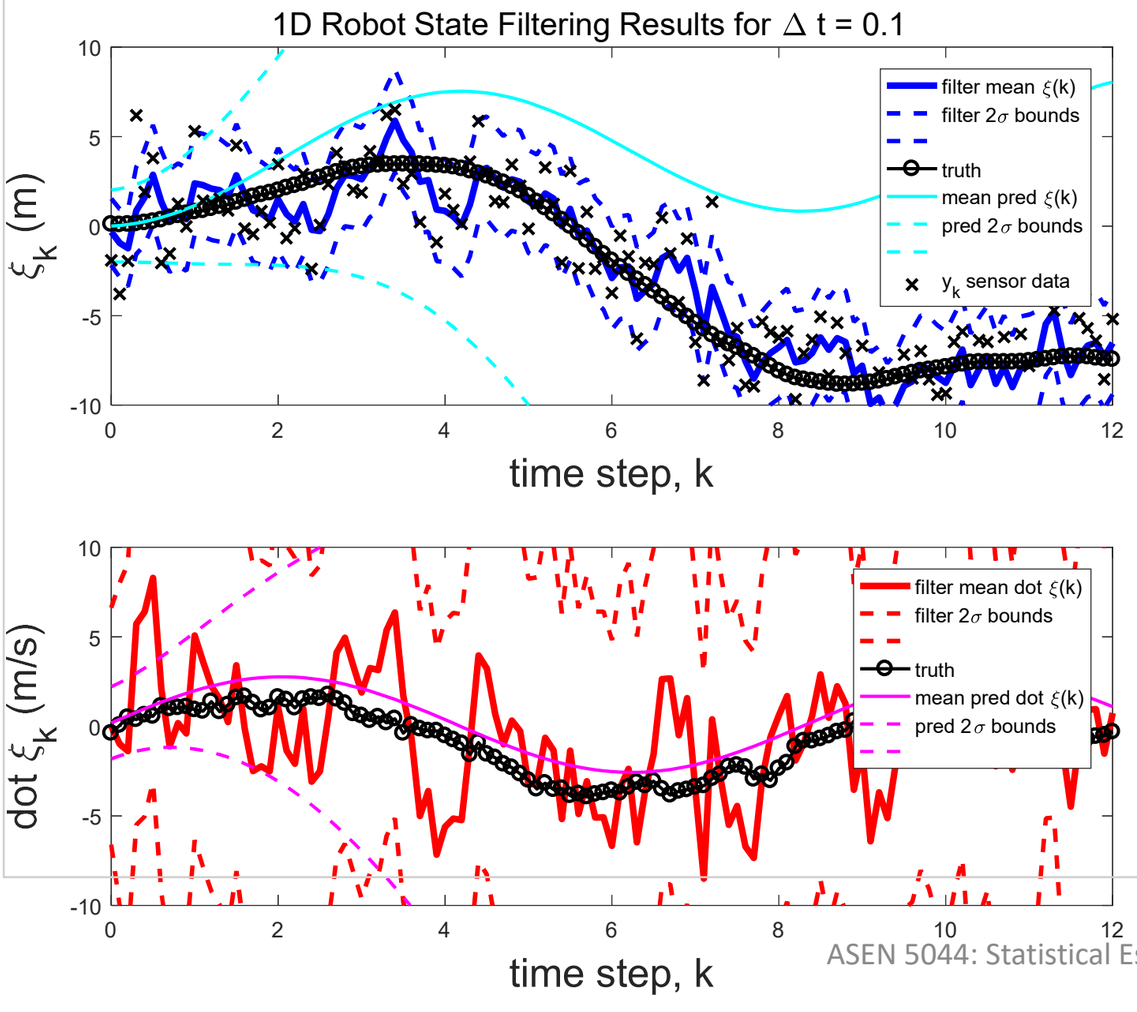


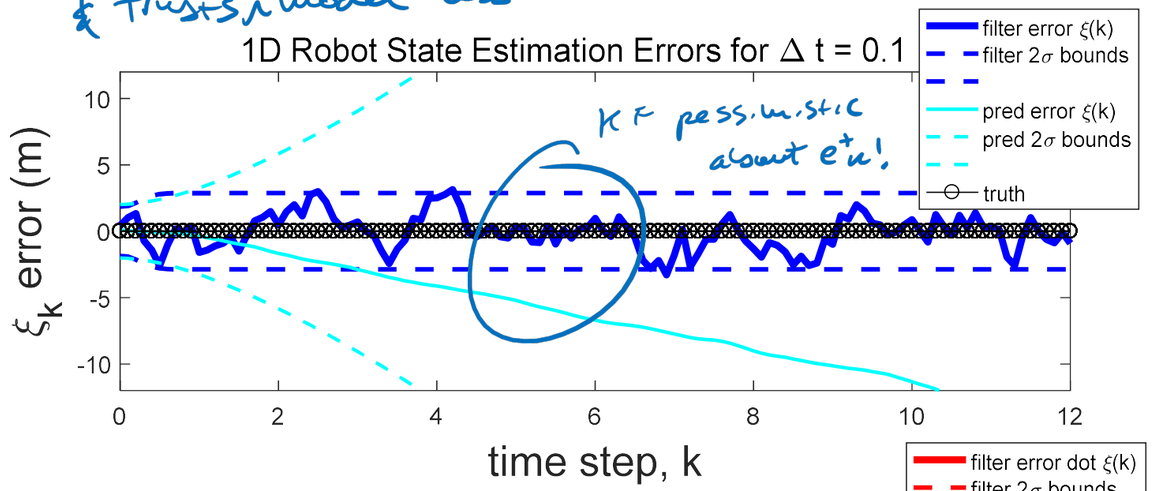


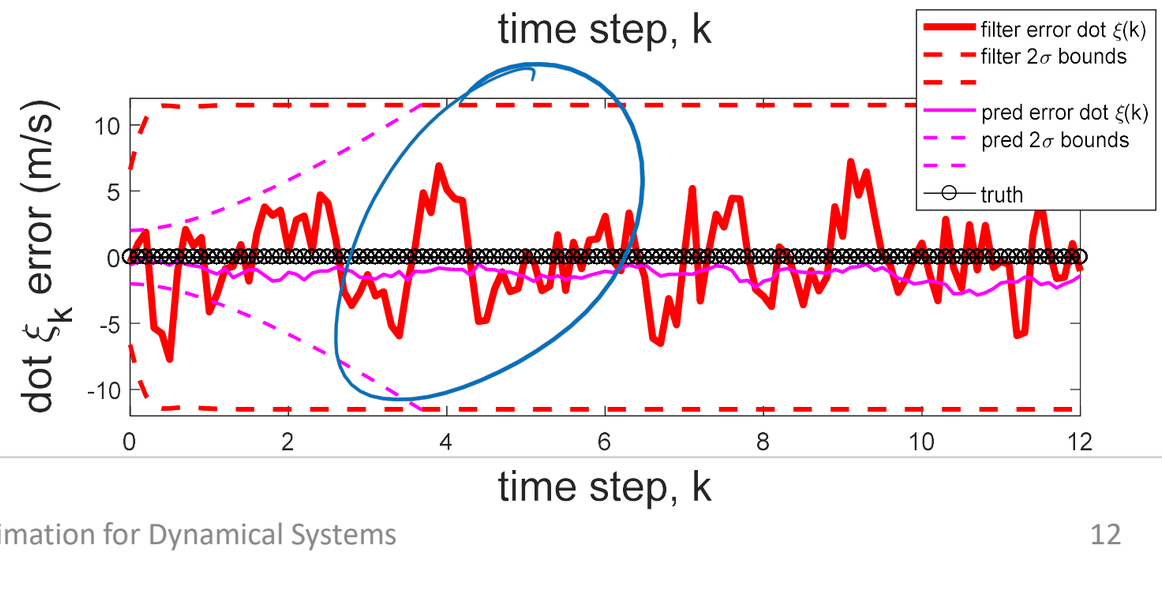
KF Results…

- See Matlab code: Lec26_1Drobotstatefilter.m
- What happens if our KF uses the incorrect values for the process noise covar Q?
- Example: now suppose Q_{KF} = 100* Q

→ Filter tuning trusts meas. y_k too much! & trusts model less than it should } KF gain too high







KF Results…

- See Matlab code: Lec26_1Drobotstatefilter.m
- What if the KF "thinks" we have a worse lidar sensor, e.g. if R_{KF} = 10*R?

  (i.e. the KF doesn't "trust" the sensors as much as it would otherwise)