ASEN 5044, Fall 2018

Statistical Estimation for Dynamical Systems

Lecture 30:
Introduction to Nonlinear Filtering;
Jacobians for DT Nonlinear Filtering

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

Tues 11/27/2018

University of Colorado Boulder

Announcements

- Midterm 2 grading should be done by this weekend
- **Final project partners now finalized (no changes possible from now on)**
- **Next steps for HW 8 and final project:**
  - **HW 8 [due Tues 12/4]** = group assignment (do a KF on common system, then pick your final project system and do some initial stuff with it)
  - Final project report [**to be due Tues 12/18**] = non-linear filtering and analysis
  - Candidate systems posted

Overview

Last Time

- How to tell if your (linear) KF is actually working correctly???
- KF dynamic consistency analysis and "Truth Model Testing" (TMT)
- Chi-square tests (NEES/NIS) – check if KF's state errors/measurement residuals make sense for given system + measurement + noise models
    - Do actual state errors/meas. residuals agree with KF's estimated error covariances?
    - Formal statistical tests to examine this question

Today

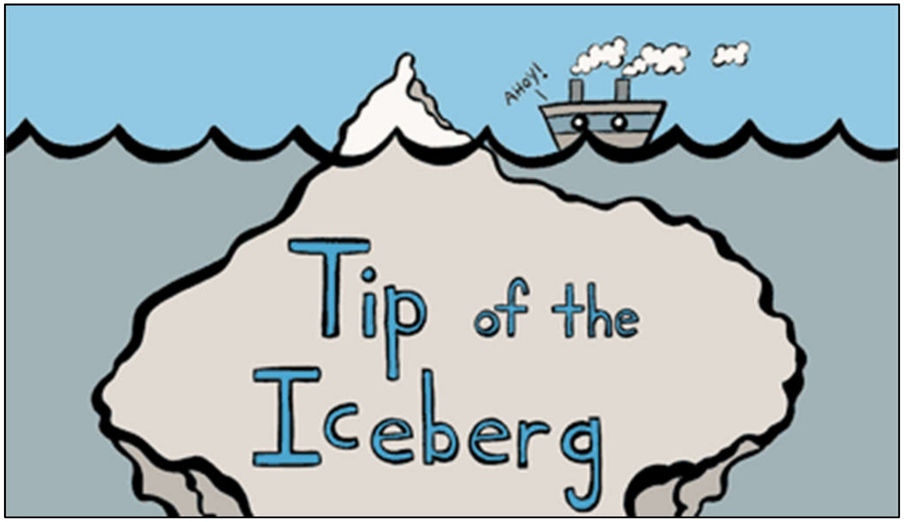
- **Intro to nonlinear dynamical state estimation (discrete time)**
    - **Optimal non-linear state estimation problem definition and setup**
- **Two popular sub-optimal "analytic" approximations**
    - **Linearized KF**
    - **Extended KF (EKF)**

READ SIMON BOOK, CHAPTERS 13.1-13.2 (nonlinear KFs)

Introduction to Nonlinear Estimation

Roadmap for remaining few lectures:

- To what extent **do linear estimation methods apply to non-linear problems**?
- Basic but widely used methods based on **linearization**:
    - Linearized KF
    - Extended Kalman Filter (EKF)
    - Unscented Kalman Filter (UKF)



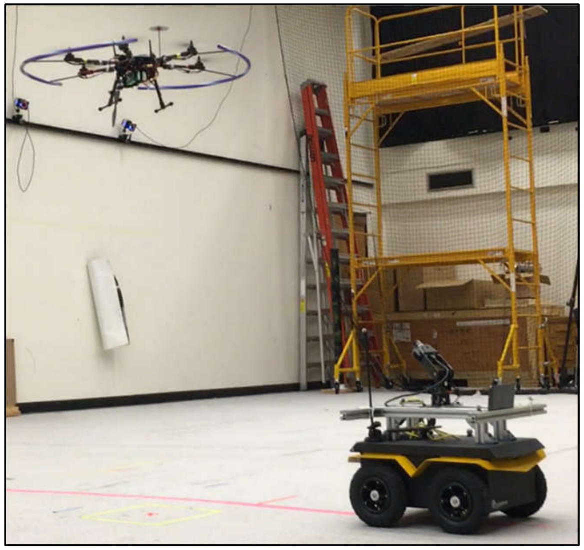
- Survey more powerful/general methods:
    - Nonlinear least squares (NLS)
    - Maximum likelihood
    - Bayesian filters and estimators

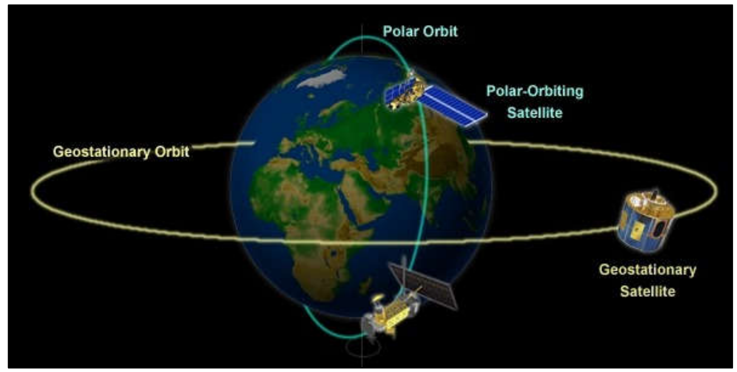
    (particle filters, Gaussian mixture KFs, …)

    [advanced classes]

Example Applications: Final Project Systems

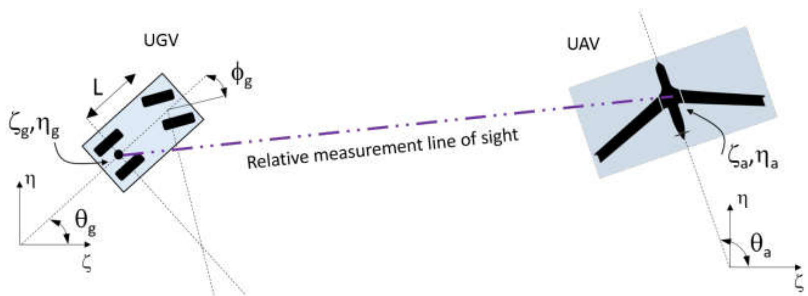
- You will be quite familiar with a non-linear filtering problem by end of semester…





Example Applications: Final Project Systems

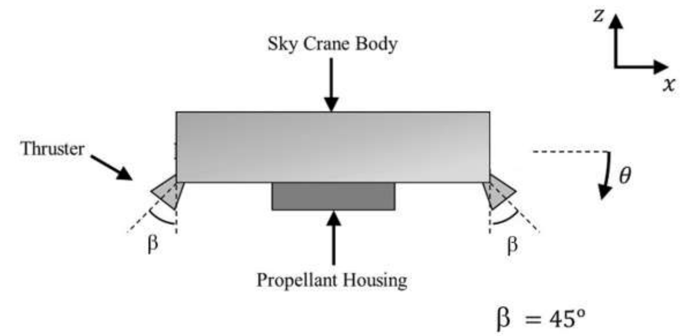
- You will be quite familiar with a non-linear filtering problem by end of semester…



$$
\begin{aligned}
\dot{\zeta}_g &= v_g \cos\theta_g + \tilde{w}_{x,g} \\
\dot{\eta}_g &= v_g \sin\theta_g + \tilde{w}_{y,g} \\
\dot{\theta}_g &= \frac{v_g}{L}\tan\phi_g + \tilde{w}_{\omega,g},
\end{aligned}
$$

$$
\begin{aligned}
\dot{\zeta}_a &= v_a \cos\theta_a + \tilde{w}_{x,a} \\
\dot{\eta}_a &= v_a \sin\theta_a + \tilde{w}_{y,a} \\
\dot{\theta}_a &= \omega_a + \tilde{w}_{\omega,a}
\end{aligned}
$$

$$
\mathbf{y}(t) = \begin{bmatrix} \tan^{-1}\left(\frac{\eta_a-\eta_g}{\xi_a-\xi_g}\right) - \theta_g \\ \sqrt{(\xi_g-\xi_a)^2 + (\eta_g-\eta_a)^2} \\ \tan^{-1}\left(\frac{\eta_g-\eta_a}{\xi_g-\xi_a}\right) - \theta_a \\ \xi_a \\ \eta_a \end{bmatrix} + \tilde{\mathbf{v}}(t),
$$



$$
\begin{aligned}
\ddot{\xi} &= \frac{[T_1(\cos\beta\sin\theta + \sin\beta\cos\theta) + T_2(\cos\beta\sin\theta - \sin\beta\cos\theta) - F_{D,\xi}]}{m_b + m_f} + \tilde{w}_1, \\
\ddot{z} &= \frac{[T_1(\cos\beta\cos\theta - \sin\beta\sin\theta) + T_2(\cos\beta\cos\theta + \sin\beta\sin\theta) - F_{D,z}]}{m_b + m_f} - g + \tilde{w}_2, \\
\ddot{\theta} &= \frac{1}{I_\eta}\left[(T_1 - T_2)\cos\beta \cdot \frac{w_b}{2} + (T_2 - T_1)\sin\beta \cdot h_{cm}\right] + \tilde{w}_3, \\
I_\eta &= \frac{1}{12}\left[m_b(w_b^2 + h_b^2) + m_f(w_f^2 + h_f^2)\right], \\
F_{D,\xi} &= \frac{1}{2}C_D\rho[A_{side}\cos(\theta-\alpha) + A_{bot}\sin(\theta-\alpha)] \cdot \dot{\xi}\sqrt{\dot{\xi}^2 + \dot{z}^2}, \\
F_{D,z} &= \frac{1}{2}C_D\rho[A_{side}\cos(\theta-\alpha) + A_{bot}\sin(\theta-\alpha)] \cdot \dot{z}\sqrt{\dot{\xi}^2 + \dot{z}^2}, \\
\alpha &= \tan^{-1}\left(\frac{\dot{z}}{\dot{\xi}}\right),
\end{aligned}
$$

Non-linear vs. Linear Estimation Problems

- What makes these estimation problems **“non-linear”?**
- Real problems often have non-linearities in dynamics and/or measurements
- <u>Dynamics</u>: solutions to EOM/ODEs <u>do not</u> obey superposition
    - cannot just look at deterministic and random inputs separately
    - no simple closed-form solutions or behaviors (tricky to analyze)
- <u>Measurements</u>: complex relationships to states
    - difficult to “invert” to get state info from sensor data (<u>observability analysis hard</u>)
- Process/sensor noises not necessarily additive (or Gaussian)

- Final project systems all have “smooth” nicely differentiable non-linearities
- Can get “non-smooth” types: e.g. saturation, hysteresis, angle wrap, discrete switches,…
- Linear estimation methods often adapted to “smooth” cases via linearization

  → approx to optimal LS filters: many caveats and no guarantees!!! (but generally still work fine)

(Semi-)formal problem statement…

- How to define an "optimal" state estimator for a nonlinear dynamical system?

$x(t) \in \mathbb{R}^n$

$y(t) \in \mathbb{R}^p$

$$\dot{x}(t) = \mathcal{F}[x(t), u(t), \tilde{w}(t)]$$
$$y(t) = \mathcal{H}[x(t), \tilde{v}(t)]$$

$\mathcal{F}$: nonlinear CT dyn. fxn

$\mathcal{H}$: " " meas. fxn

$$x(k+1) = f[x(k), u(k), w(k)], \qquad w(k) = \mathcal{N}(0, Q) \text{ (AWGN)}$$
$$y(k+1) = h[x(k+1), v(k+1)], \qquad v(k) = \mathcal{N}(0, R) \text{ (AWGN)}$$

- Follow same logic as before with linear systems to set up a cost fxn J(K)in DT:

$$\text{let } \; e_k^+ = x_k - \hat{x}_k^+,$$

$$J(k) = E[e_k^{+T} e_k^+] = \text{trace}(E[e_k^+ e_k^{+T}]) = \text{trace}(E[e_k^+ e_k^{+T}]) = \text{trace}(P_k^+)$$

FACT: it is possibly to show that, generally, $J(k)$ is minimized by:

$$\hat{x}_k^+ = E[x_k | y_{1:k}]_{p(x_k|y_{1:k})} \text{ (conditional mean of } x_k \text{ given all data } y_1, \ldots, y_k)$$

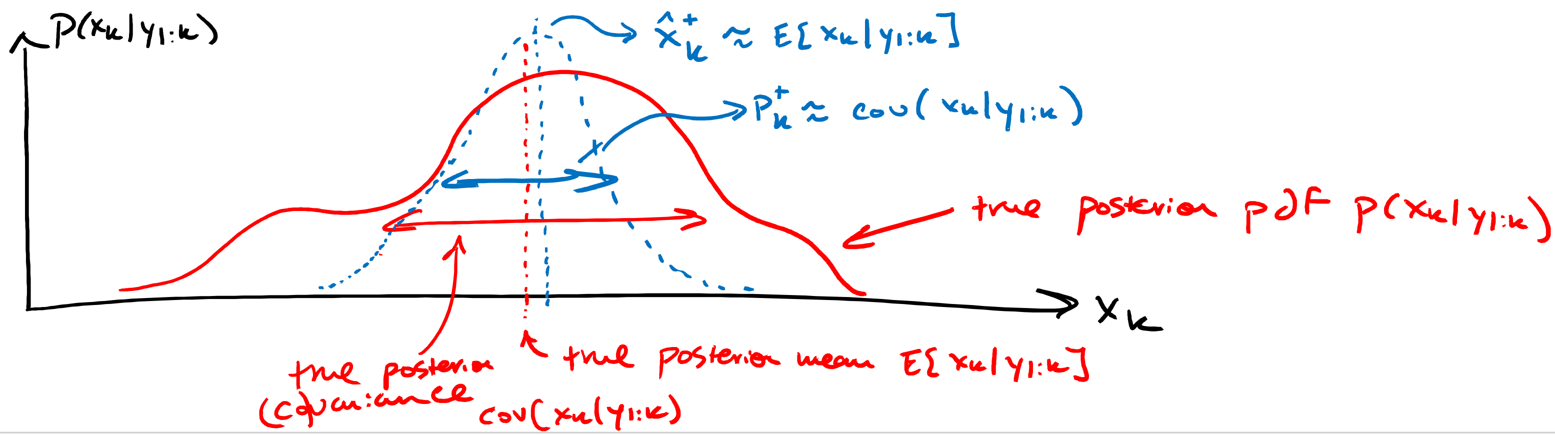
Issues with the "Exact" Optimal Estimator

- Theoretically works for <u>any</u> set of dynamics/measurements models → only need to get the posterior pdf $p(x_k \mid y_{1:k})$ and read off its mean and covariance!!

The KF for linear-Gaussian systems computes exactly
the posterior mean $\longleftrightarrow$ $\hat{x}_k^+ = E[x_k \mid y_{1:k}]$
& the posterior covariance $\longleftrightarrow$ $P_k^+ = \text{cov}[x_k \mid y_{1:k}]$

Proof: see adv. topic lectures 25 + 28

- But practically computing/representing posterior pdf is also <u>very hard</u> in theory and in practice for non-linear/non-Gaussian problems!



Approximating the Optimal Estimator

Most popular workaround:

- if sample time ΔT is not "too big"…
- and if nonlinearities are "smooth enough"…

→then can use DT linearization to get approx. optimal solutions from a linear KF

→this approximately tracks posterior pdf $p(x_k|y_{1:k})$ mean and covariance

(don't need full posterior pdf -- just recursively update the first two moments!!)

- Key trick: use given nonlinear CT model to get a "proxy" linearized DT model about some nominal state trajectory to define KF updates

$$\dot{x}(t) = \mathcal{F}[x(t), u(t), w(t)] \xrightarrow{\Delta T} x(k+1) = f[x(k), u(k), w(k)] \approx x_{\text{nom},k} + \tilde{F}_k \delta x_k + \tilde{G}_k \delta u_k + \tilde{\Omega}_k w_k$$

$$y(t) = \mathcal{H}[x(t), v(t)] \longrightarrow y(k+1) = h[x(k+1), v(k+1)] \approx y_{\text{nom},k+1} + \tilde{H}_{k+1} \delta x_{k+1} + v_{k+1}$$

Two Subtle Mathematical Issues

- #1: Linearization of CT nonlinear system in general leads to LTV approximation
- Example: consider a 2D robot with a "Dubin's unicycle" model:

$$\begin{bmatrix} \dot{\zeta}_a = v_a \cos\theta_a + \tilde{w}_{x,a} \\ \dot{\eta}_a = v_a \sin\theta_a + \tilde{w}_{y,a} \\ \dot{\theta}_a = \omega_a + \tilde{w}_{\omega,a} \end{bmatrix} = \dot{x} = \mathcal{F}[x, u, \tilde{w}],$$

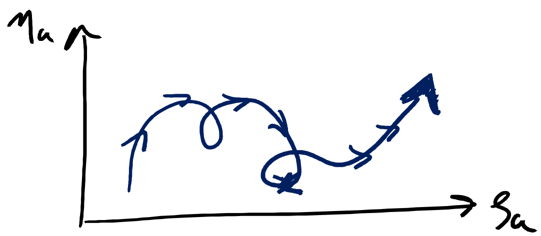
$$x(t) = \begin{bmatrix} \xi_a \\ \eta_a \\ \theta_a \end{bmatrix}, \quad u(t) = \begin{bmatrix} v_a \\ \omega_a \end{bmatrix}, \quad \tilde{w}(t) = \begin{bmatrix} \tilde{w}_{x,a} \\ \tilde{w}_{y,a} \\ \tilde{w}_{\omega,a} \end{bmatrix}$$

CT Jacobians w.r.t. state vars

$$\frac{\partial \mathcal{F}}{\partial x(t)} = \begin{bmatrix} 0 & 0 & -v_a(t)\sin\theta_a(t) \\ 0 & 0 & v_a(t)\cos\theta_a(t) \\ 0 & 0 & 0 \end{bmatrix}$$

NOT TIME INVARIANT!

Depends on state & input @ time t!



- If we can discretize and then linearize a CT nonlinear model about a time-varying state trajectory, then this generally yields LTV DT model with time-varying $(\tilde{F}_k, \tilde{G}_k, \tilde{\Omega}_k, \tilde{H}_k)$
- BUT: **will linear KF ideas still work for LTV dynamics models???**

Useful Fact #1: The Linear KF for LTV Systems

- The linear KF naturally extends to LTV DT systems <u>as long as the matrices only depend on time</u> (i.e. not also depend on state/inputs)

If *actual* dynamics are truly LTV:

$$x(k+1) = F_k x_k + G_k u_k + \Omega_k w_k$$

$$y(k+1) = H_{k+1} x_{k+1} + v_{k+1}$$

↳ can have p(# meas.) also be time varying (as long as n = # states fixed)

KF Time update/Prediction

$$\hat{x}^-_{k+1} = F_k \hat{x}^+_k + G_k u_k$$

$$P^-_{k+1} = F_k P^+_k F^T_k + \Omega_k Q \Omega^T_k$$

KF Meas update/Correction

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

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

$$K_{k+1} = P^-_{k+1} H^T_{k+1} [S_{k+1}]^{-1}$$

- But, for nonlinear filtering problem, we want to use Jacobians that must be evaluated along state trajectories – <u>so there is a state dependence!</u>

- But we can "cheat" by linearizing around "known nominal trajectory" (solution to nonlinear ODE), <u>so that we can *pretend* it is only a time-varying dependence</u>

$$\Rightarrow (\tilde{F}_k, \tilde{G}_k, \tilde{\Omega}_k, \tilde{H}_k)$$

- So will basically need to cross our fingers and hope that nominal trajectory stays "close enough" to what system actually doing! (hence: no formal guarantees for the linearized KF/EKF…)

Two Subtle Mathematical Issues

- #2: How to get a CT nonlinear model in DT and then linearize it anyway???

➔ tricky/very annoying to exactly find DT Jacobians for $\tilde{F}_k, \tilde{G}_k, \tilde{\Omega}_k$ $\quad (\tilde{H}_k \text{ is easy})$

$$\dot{x}(t) = \mathcal{F}[x(t), u(t), w(t)] \longrightarrow x(k+1) = f[x(k), u(k), w(k)] \approx x_{nom}(k+1) + \tilde{F}_k \delta x_k + \tilde{G}_k \delta u_k + \tilde{\Omega}_k w_k$$

ΔT

$$y(t) = \mathcal{H}[x(t), v(t)] \longrightarrow y(k+1) = h[x(k+1), v(k+1)] \approx y_{nom}(k) + \tilde{H}_{k+1} \delta x_{k+1} + v_{k+1}$$

$$\tilde{F}_k = \frac{\partial f}{\partial x_k}\Big|_{nom[k]} \qquad \tilde{G}_k = \frac{\partial f}{\partial u_k}\Big|_{nom[k]} \qquad \tilde{\Omega}_k = \frac{\partial f}{\partial w_k}\Big|_{nom[k]} \qquad \tilde{H}_k = \frac{\partial h}{\partial x_k}\Big|_{nom[k]}$$

$(\rightarrow \text{easy since } h = \mathcal{H}!)$

$f[x_k, u_k, w_k]$ generally not closed form $\rightarrow$ DT f Jacobians not closed form !!!

$\rightarrow$ **DT Jacobians must be computed numerically**

(this is one reason some people don't like using linearized KFs/EKFs at all!)

➔Fortunately, a simple approximation procedure <u>based on CT Jacobians</u> works reasonably well for linearized KF/EKF calculations when ΔT sufficiently small…

Useful Fact #2: "Eulerized" DT Jacobians

- Use Euler integration to approximate DT state transition fxn for small ΔT
- Then take partial derivatives of this to approximate required DT Jacobians
- Naturally get to use CT Jacobians as part of result

Start with (mild) assumption that the CT nonlinear model can be generally written as

$$\dot{x}(t) = \mathcal{F}[x(t), u(t)] + \Gamma(t) \cdot \tilde{w}(t)$$

Euler approx: $x(t_{k+1}) \approx x(t_k) + \Delta T \cdot \dot{x}(t)\big|_{t=t_k}$

$\longrightarrow \quad x_{k+1} \approx x_k + \Delta T \cdot \dot{x}(t=t_k)$

$= x_k + \Delta T \cdot \{ \mathcal{F}[x(t_k), u(t_k)] + \Gamma(t_k) \cdot \tilde{w}(t_k) \} \approx \underline{f}(x_k, u_k, w_k)$

$\longrightarrow$ Then $\tilde{F}_k = \frac{\partial f}{\partial x_k}\Big|_{nom[k]} = \frac{\partial x_{k+1}}{\partial x_k}\Big|_{nom[k]} = \frac{\partial}{\partial x_k}\Big(x_k + \Delta T \cdot \{ \mathcal{F}[x(t_k), u(t_k)] + \Gamma(t_k) \cdot \tilde{w}(t_k) \} \Big)\Big|_{nom[k]}$

$= \frac{\partial}{\partial x_k}(x_k)\Big|_{nom[k]} + \frac{\partial}{\partial x_k}\{ \Delta T \cdot \mathcal{F}[\cdots] + \Gamma(t_k) \cdot \tilde{w}(t_k) \}\Big|_{nom[k]} = I + \Delta T \cdot \frac{\partial \mathcal{F}[\cdots]}{\partial x_k}\Big|_{t=t_k, nom[k]}$ $\longrightarrow$ CT Jacobian matrix!

$\boxed{= I + \Delta T \cdot \tilde{A}\big|_{nom[k]}}$ (*)