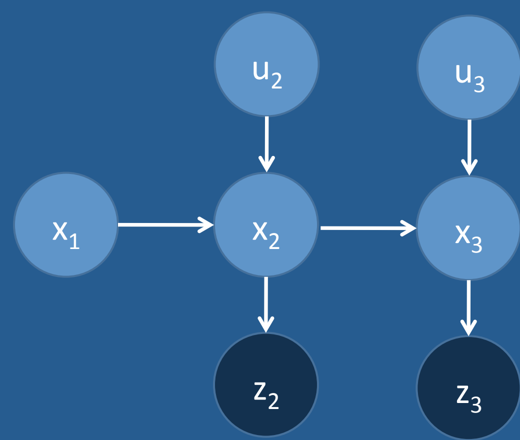
1



L20. STATE ESTIMATION

EECS 492
March 22, 2011

State Estimation

2

- State Vector:
  - Nx1 column vector of quantities we care about.

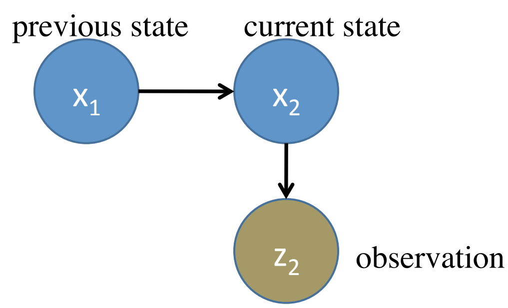
$$x = \begin{bmatrix} x \\ y \\ \theta \\ s \end{bmatrix}$$

  - Which quantities to include is an engineering choice
  - Could also estimate acceleration, angular velocity.
  - Could also include information about the world (e.g. landmarks)
- State Estimation:
  - The probabilistic estimation of the state vector.

State Estimation

3

- Our graphical model



- Many ways of representing probability distribution
  - We'll use multivariate Gaussians

State Estimation: Overview

4

- Suppose at time step 1, we have an estimate of our state vector (our *prior*):

$$p(x_1)$$

- Two basic operations:
  - Propagation
    - Account for passage of time
  - Observation
    - Incorporate information from sensors

Propagation

5

- Suppose that time Δt passes. How does our state evolve?
  - Some function of our state x and noise w:

$$\begin{bmatrix} x' \\ y' \\ \theta' \\ s' \end{bmatrix} = f(x, w) = \begin{bmatrix} x + s\Delta t\cos(\theta) + w_1 \\ y + s\Delta t\sin(\theta) + w_2 \\ \theta + w_3 \\ s + w_4 \end{bmatrix}$$

- How do we update our mean and covariance?
  - Covariance projection!

Propagation

6

$$u_x' = f(u_x, E(w))$$

- Propagate mean?
  - Just plug in current state value.
  - Usually, E(w) = 0
- Propagate covariance?
  - It's non-linear, so linearize.
  - But propagation is function of state and w…
    - Linearize WRT both!

Propagation

7

- Linearize: $\begin{bmatrix} x' \\ y' \\ \theta' \\ s' \end{bmatrix} = f(x,w) = \begin{bmatrix} x + s\Delta t\cos(\theta) + w_1 \\ y + s\Delta t\sin(\theta) + w_2 \\ \theta + w_3 \\ s + w_4 \end{bmatrix}$

$$f(x,w) \approx J_x^f(x - u_x) + J_w^f(w - u_w) + f(u_x, u_w)$$

- Our Jacobians:

$$J_x^f = \begin{bmatrix} 1 & 0 & -s\Delta t\sin(\theta) & \Delta t\cos(\theta) \\ 0 & 1 & s\Delta t\cos(\theta) & \Delta t\sin(\theta) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad J_w^f = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Propagation (summary)

8

- Write down propagation equation in terms of previous state and noise:

$$x' = f(x,w)$$

- Linearize by computing Jacobians J_x^f, J_w^f

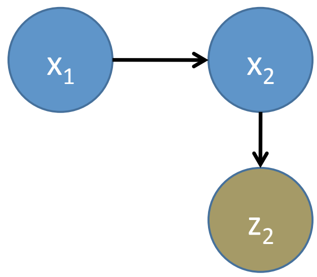
- Propagate:

$$u'_x = f(u_x, u_w) = f(u_x, 0)$$
$$\Sigma'_x = J_x^f \Sigma_x J_x^{f\,T} + J_w^f \Sigma_w J_w^{f\,T}$$

Observation

9

- Suppose we get a sensor observation:



- The observation tells us something about our state. What distribution do we want now?
  - We want the state given all data (observations)!

$$p(x_2|z_2)$$

What is this in terms of quantities that we know???

Observation

10

- We want: $p(x_2|z_2)$
- Apply Bayes' rule:

$$p(x_2|z_2) = \frac{\overbrace{p(z_2|x_2)}^{\textit{sensor model}}\overbrace{p(x_2)}^{\textit{prior from propagation step}}}{\underbrace{p(z_2)}_{\textit{normalization constant}}}$$

- How do we get these quantities?

Sensor Model

11

- Perhaps we have a compass that observes the heading, contaminated by white noise w_1

$$z(x) = \theta + w_1$$

- If we know how w_1 is distributed, we can compute the distribution $p(z|x)$
  - Mean and covariance projection again!

Observation: Putting things together

12

- We want our posterior distribution
  - Condition on evidence

$$p(x_2|z_2) \propto p(z_2|x_2)p(x_2)$$

- We're representing each of these probabilities as Gaussian random variables, so we can write:

$$p(x_2|z_2) \propto Ke^{-\frac{1}{2}(z_2-z(x))^T\Sigma_z^{-1}(z_2-z(x))}e^{-\frac{1}{2}(x-u_x)^T\Sigma_x^{-1}(x-u_x)}$$

Observation: Putting things together

13

$$p(x_2|z_2) \propto K e^{-\frac{1}{2}(z_2 - z(x))^T \Sigma_z^{-1}(z_2 - z(x))} e^{-\frac{1}{2}(x-u_x)^T\Sigma_x^{-1}(x-u_x)}$$

Observed value

Predicted value

Linearization: H = Jacobian

$$\begin{aligned} z - z(x) &= z - Hd - z_0 \\ &= r - Hd \end{aligned}$$

- We want to write the posterior as a Gaussian.
  - What are the parameters of that Gaussian?
  - Note: mean of covariance is its maximum!
- Substitute r = z – z0 and take logarithm:

Note the structure!

$$\chi^2 = (r - Hd)^T \Sigma_z^{-1}(r - Hd) + d^T \Sigma_x^{-1} d$$

A little math…

14

$$\chi^2 = (r - Hd)^T \Sigma_z^{-1}(r - Hd) + d^T \Sigma_x^{-1} d$$

- Expand…

$$\chi^2 = r^T\Sigma_z^{-1} r - 2d^T H^T \Sigma_z^{-1} r + d^T H^T \Sigma_z^{-1} H d + d^T \Sigma_x^{-1} d$$

- Minimize by differentiating WRT d:

$$\frac{\partial \chi^2}{\partial d} = -2H^T\Sigma_z^{-1} r + 2H^T\Sigma_z^{-1} H d + 2\Sigma_x^{-1} d = 0$$

$$d = (H^T\Sigma_z^{-1} H + \Sigma_x^{-1})^{-1} H^T \Sigma_z^{-1} r$$

A solution, at last!

15

$$d = (H^T \Sigma_z^{-1} H + \Sigma_x^{-1})^{-1} H^T \Sigma_z^{-1} r$$
$$x = x + d$$

- Computational complexity?
  - Matrix inversion is $O(N^3)$ and N is the dimension of the whole state vector!
- Memory requirements?
  - We're going to have to store the covariance matrix, which is $O(N^2)$

Improving the method

16

- Matrix inversion lemma (for invertible C):

$$(A + BCD)^{-1} BC = A^{-1} B (C^{-1} + DA^{-1} B)^{-1}$$

http://www.cs.ucl.ac.uk/staff/G.Ridgway/mil/mil.pdf

$$d = (H^T \Sigma_z^{-1} H + \Sigma_x^{-1})^{-1} H^T \Sigma_z^{-1} r$$

$$\Downarrow$$

$$d = \Sigma_x H^T (\Sigma_z + H \Sigma_x H^T)^{-1} r$$

- Computational complexity now?

Extended Kalman Filter

- This method called EKF
  - We've glossed over the derivation for covariance updates for observation… they're ugly.

- Slightly more general/standard form:

$$K = \Sigma_x^- J_x^T (J_x \Sigma_x^- J_x^T + J_w \Sigma_w J_w^T)^{-1}$$
$$x = x^- + K(z - f(x^-, 0))$$
$$\Sigma_x = \Sigma_x^- - K J_x \Sigma_x^-$$

EKF: Intuition

- It's a low pass filter

$$x = x^- + K(z - f(x^-, 0))$$

How much does the observation disagree with our prior? **"innovation"**

How much do we trust this measurement, and should we adjust our state? **"Kalman gain"**

A mixture of our previous estimate and the observation.

- Compare to IIR filter: y[n] = y[n-1] + *a*x[n]
  - EKF: we adjust gain *a* at every iteration

EKF: Intuition (Cartoon version!)

$$K = \Sigma_x^- J_x^T (J_x \Sigma_x^- J_x^T + J_w \Sigma_w J_w^T)^{-1}$$
$$x = x^- + K(z - f(x^-, 0))$$

(Pretend that J is invertible)

$$x = x^- + J_x^{-1} \frac{J_x \Sigma_x^- J_x^T}{J_x \Sigma_x^- J_x^T + J_w \Sigma_w J_w^T} (z - f(x^-, 0))$$

What fraction of the uncertainty was our prior responsible for? (larger → trust observation more)

How much does the observation disagree with our prior?

Project from observation space to state space

EKF: Optimal *Recursive* Estimator

20

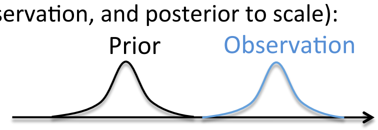
- Why do we call it a *recursive* estimator?
- Suppose I get a stream of observations
  - p(z1 | x), p(z2|x), p(z3|x), ….

  p(x | z1, z2, z3) = α p(z3 | x) p(x | z1, z2)
  p(x | z1, z2) = α p(z2 | x) p(x | z1)
  p(x | z1) = α p(z1 | x) p(x)
- We only have to maintain p(x | …)
  - Can incorporate new evidence without remembering old evidence.

Mini Quiz: Your turn!

21

1. T/F: Covariance projection can only increase uncertainty, |P|
2. T/F: Observations can only decrease uncertainty
3. Sketch the posterior resulting from the prior and observation below (draw the prior, observation, and posterior to scale):



4. T/F: A large Kalman gain results from a *strong* prior and a *weak* observation.
5. T/F: The Kalman filter is a sound inference method for linear problems.
6. Suppose we are tracking tanks from an airplane, estimating x, y, and velocity. What is the effect on time and memory if we *double* the number of tanks we're tracking?

Why "Extended" Kalman Filter?

- The "Kalman Filter" was originally derived for purely linear systems
- Applicability is thus limited to linear problems
- Extended Kalman Filter is generalization to non-linear systems
  - But inexact!

EKF: Linearization Error

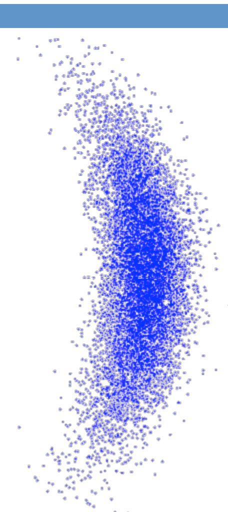
- Observations are "incorporated" only once
  - State and covariance are updated based on linearization point *at that point in time*

- If state estimate is inaccurate, linearization point will be inaccurate.
  - Introduces error into state estimate
  - Covariance is decreased as though there was no error introduced.
  - Filter becomes over-confident.

EKF: Linearization Error

24

- Nonlinear problems have non-linear uncertainties
  - (theta noise greatly exaggerated)

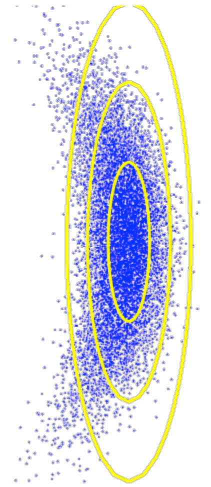


- This shape cannot be represented by a Gaussian distribution.
  - So what happens?

EKF: Linearization Error

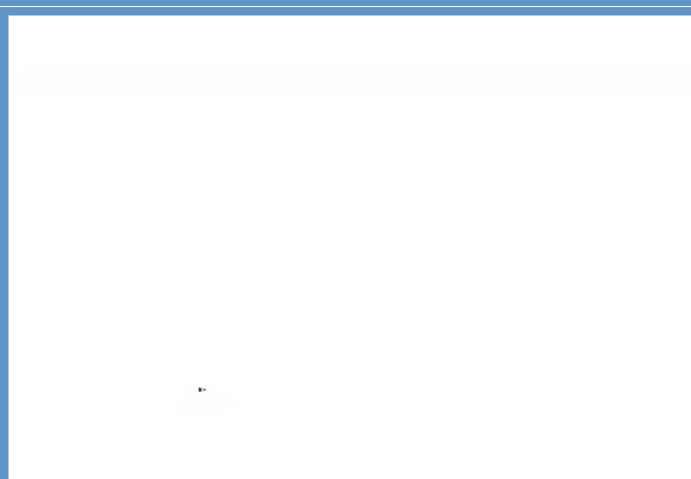
25

- Mean and covariance are computed around the expected value
- Non-linear behavior away from expected value is not well approximated.
- Could we improve upon this?
  - Unscented Kalman Filter
  - Particle Filter



Demo: No Observations

26



Demo: With observations

27

Review

28

Next Time

29

- Decision Processes