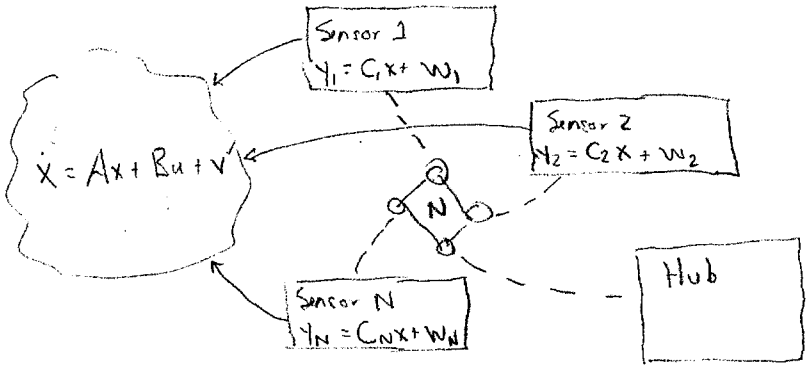
Ref: Olfati-Saber
CDC 2007

RMM 19 Mar 08

Distributed Sensor Fusion



Consider a single process with multiple sensors, networked together
Would like to form estimate $\hat{x}$, either at each sensor or a central hub

Examples;

1. Alia - multiple sensors looking at environment plus possible need for different information at different points (eg. urban planning)
2. RoboFlag - each robot needs estimate of (local) environment plus players need to know entire environment

Case 1: centralized hub

Easy approach: sensors send data to hub, the hub runs (large) KF
Alternative approach: information filter (covered by Ling)

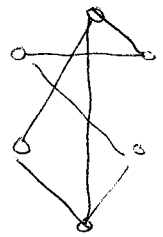
~~- Start with static case (no dynamics)~~

History: Chong (1979) - Hierarchical KF, Willsky et al (1982)

Case 2: fully connected network

Everyone sends either measurements or estimates to each other
Reconstruct state at each node (ala info filter)

Case 3: distributed information on a graph



Suppose we have no central hub and we want each sensor to converge to a global estimate?

Static case: $\hat{x}^i(\infty) = \sum_{j=1}^{N} R_j^{-1} y^j$

Starting point: consensus algorithms

Let $G = (V, E)$ be a graph describing the information flow.
Write $\mathcal{N}_i = \{v_{i_1}, v_{i_2}, \dots, v_{i_{m_i}}\}$ for the neighbors of agent i.

Consensus "protocol"

$$\dot{x}^i = \frac{1}{|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} (x^j - x^i) = -(x^i - \text{Avg}(x^{\mathcal{N}_i}))$$

$$= -\tilde{L}x \qquad \tilde{L} = D^{-1}(D - A)$$

$\leftarrow$ weighted Laplacian

D: degree matrix; A: adjacency matrix

Facts about Laplacians (& weighted Laplacian)

1. $L\mathbf{1} = 0$
2. For undirected graphs, spectrum is $0 \le \lambda_2 \le \lambda_3 \le \dots \le \lambda_N$
3. $\lambda_2 \neq 0 \Leftrightarrow$ graph is connected

Thm If G is connected, $x \to \alpha \mathbf{1} \Rightarrow$ consensus and $\alpha = \frac{1}{N} \sum x^i(0) = \text{Avg}(x(0))$

Pf Need to show two things: 1) $x \to \alpha 1$ and 2) $\alpha = Avg(x(0))$

1) Let $\delta = x - Ave(x(0))\,1$ = disagreement vector. $\dot{\delta} = L\delta$

Set $V = \delta^T\delta$ and compute $\dot{V}$

$$\dot{V} = -2\delta^T\dot{x} = -2\delta^T L\delta$$

$\dot{V} = 0 \Rightarrow \delta = \tilde{\alpha} 1 \Rightarrow x = \alpha 1 \Rightarrow$ asy consensus

2) Let $W(x) = \frac{1}{N}\sum x(t) = Avg(x(t)) = 1^T x(t)$

$$\dot{W} = \frac{1}{N} 1^T \dot{x} = \frac{1}{N} 1^T L x = 0 \quad (L = L^T)$$

$\Rightarrow W$ is a conserved quantity. $W(x(t)) = W(x(0))$

and so $Avg(x(t)) = Avg(x(0)) \Rightarrow \alpha = Avg(x(0))$

Application: Use consensus to solve distributed averaging

- Each node measures y^i
- Set $x(0) = R_i^{-1} y^i$ (R_i = covariance for sensor i)
- Run consensus protocol $\Rightarrow x^i(\infty) = \sum_{j=1}^{N} R_j^{-1} y^j$

Remarks

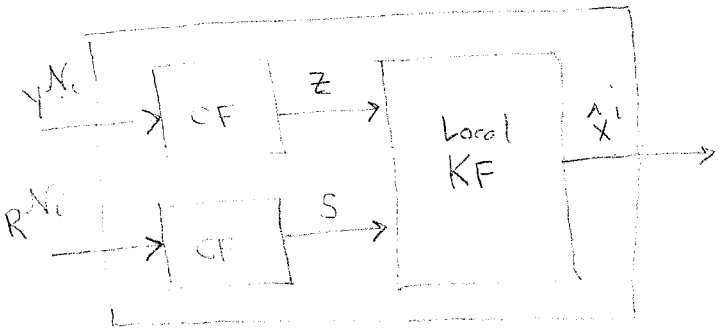
1. Can show that convergence rate is bounded by $e^{-\lambda_2 t}$
2. Discrete time version also possible
3. Lots of extensions: time-varying graphs, delays, intermittent

Extending to (dynamic) Kalman filter

Approach #1: Fixed graph (Denial, White et al)

- Communicate measurements & run full KF
- Communicate local estimates, covariance & account for duplication ( )
- Doesn't handle dropped packets, changing comms graph

Approach #2: "Microfilter Architecture" (Olfati-Saber + Sparrow)



$$\text{CF}\begin{cases} z(k) = \frac{1}{n}\sum_{i=1}^{N} C_{(i)}^T R_{(i)}^{-1} y_i(k) \\ S(k) = \frac{1}{n}\sum_{i=1}^{n} C_i^T(k) R_i^{-1}(k) C_i(k) \end{cases}$$

$$\text{KF}\begin{cases} M_i(k) = (P_i(k) + S(k))^{-1} \\ \hat{x}(k) = \bar{x}(k) + M_i(k)\left[z_i(k) - S(k)\bar{x}\right] \\ P_i(k+1) = \cdots (M_i) \\ \bar{x}(k+1) = A_k \hat{x}(k) \end{cases}$$

Idea: create consensus on measurements and covariance, then do KF (info filter variant)

Remarks

1. Need CF to converge quickly (compared to KF dynamics) and track measurements.
2. Resulting filter is approximate (may not be optimal prior to convergence) but handles packet drops, etc (inherited from CF properties)
3. Because sending measurements + covariance matrices (if $R_i(k)$ not constant)

Approach #3: Consensus on estimates

Idea: integrate consensus update with Kalman update

$$\dot{\hat{x}}^i = A\hat{x}^i + L^i(y^i - C^i\hat{x}^i) + \gamma P^i \sum_{j\in N_i}(\hat{x}^j - \hat{x}^i) \qquad \gamma > 0$$

$$\dot{P}^i = AP^i + P^iA^T + FQF^T - L^iR^iL^{iT} \qquad L^i = P^iC^{iT}(R^i)^{-1}$$

Prop (Olfati-Saber) In the absence of noise, $\hat{x}^i \to x$.

sketch of proof

$\eta_i = x - \hat{x}^i$ $\quad V(\eta) = \sum_{i=1}^{N} \eta_i^T P_i^{-1} \eta_i$ $\quad$ Show $\dot{V} \le 0$ ala consensus.

Remarks

1. Can write discrete-time version
2. Only approximate KF; lose optimality during transient
3. Can handle varying graph, packet loss, time delay, etc
4. Only requires sending estimates; P_i is <u>local</u> error covariance (doesn't account for neighbor covariance)

Final remarks

1. Distributed KF on fixed graph (star, completely connected, undirected graph) are well understood. Moral: Relate info filter.
2. Alternative approach: use consensus filter on measurements or estimates. Lose optimality, but can handle network effects,
3. Choice of graph still partially open (eg, graphs w/ cycles)