Plan for remaining classes

12. Distributed estimation (16/5)
13. I/o networks (23/5)
14. I/o networks (25/5)
15. Reviews (30/5)
16. UAS (8/6 or 15/6 or sesuai jadwal dept)

EE185523

Distributed Estimation

Yurid E. Nugraha
TSP DTE FTEIC ITS

Distributed estimation

- Two complementary areas of distributed estimation:
  - Distributed linear least squares,
  - Distributed Kalman filtering over sensor networks
- Estimation theory: designing processes by which a static or dynamic variable of interest can be uncovered by **processing a noisy signal**.

EE185523

Distributed Estimation

Distributed linear least squares

Yurid E. Nugraha
TSP DTE FTEIC ITS

Distributed linear least squares

Linear function: $z = H\theta + v,\ \ z \in \mathbb{R}^{p}, H \in \mathbb{R}^{p\times q}$

Measurement channel

Observation matrix

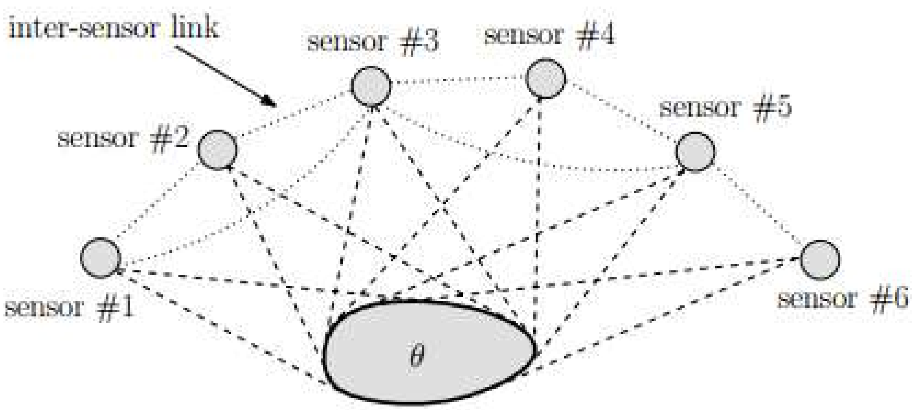
Noise

Least square estimation: Minimize $J(\theta) = (z - H\theta)^{T}(z - H\theta)$

By setting gradient to 0:

Least square estimate: $\hat{\theta} = (H^{T}H)^{-1}H^{T}z$

Least squares over sensor networks



- In a distributed setting, there are n sensors available
- Hence $z_i = H_i\theta + v_i$
- Centralized observation matrix $H = [H_1; H_2; \dots; H_n]$

- Centralized LSE then can be written as

$$\hat{\theta} = \left(\sum_{i=1}^{n} H_i^\top H_i\right)^{-1} \left(\sum_{i=1}^{n} H_i^\top z_i\right)$$

- **If** each sensors provides the raw measurement z_i to **fusion center** which has prior knowledge of H_i, then $\hat{\theta}$ can be found

Least squares over sensor networks

- However, such fusion center may not be needed in a distributed setting
- Consider iteration for ith sensor as

$$\hat{\theta}_i[k+1] = \hat{\theta}_i[k] + \Delta \sum_{j \in N_i} w_{ij}(\hat{\theta}_j[k] - \hat{\theta}_i[k]),$$

step size $\Delta \in (0,1)$

- $\hat{\theta}[0]$: prior estimate at initialization of the estimation process
- Consider edge weight $W = diag([w_1, \dots, w_m])$ for m edges

Least squares over sensor networks

Lemma 8.1

$\lim_{k\to\infty} \hat{\theta}[k] = \left(\frac{1}{n}\sum_{i=1}^{n} z_i\right)\mathbf{1}$ if and only if the network is connected and $\rho < 2/\Delta$, where ρ being largest absolute value of eigenvalue of matrix $L_w(G) = DWD^\top$, D being incidence matrix.

Design of Δ and W is important:

Suppose that the weighting diagonal matrix is designed in such a way that its jth diagonal entry associated w/ edge $j = (u, v)$ is

$$W_{jj} = (\max\{d_w(u), d_w(v)\})^{-1}.$$

Then, for any $0 \le \Delta \le 1$, then $\rho < 2/\Delta$ is satisfied.

Distributed least squares estimation: Vector

- Each sensor maintains two arrays $P_i \in \mathbb{R}^{q\times q}$ and $\hat{\theta}_i \in \mathbb{R}^q$
- Iterations:

$$P_i[k+1] = P_i[k] + \Delta \sum_{j\in N_i} w_{ij}(P_j[k] - P_i[k]),$$
$$\hat{\theta}_i[k+1] = \hat{\theta}_i[k] + \Delta \sum_{j\in N_i} w_{ij}(\hat{\theta}_j[k] - \hat{\theta}_i[k]),$$

- Initial conditions:

$$P_i[0] = H_i^\top H_i \ \ , \ \hat{\theta}_i[0] = H_i^\top z_i$$

Distributed least squares estimation: Vector

- It then follows that:

$$\lim_{k\to\infty} P_i[k] = \frac{1}{n}\sum_{i=1}^{n} H_i^\top H_i$$

$$\lim_{k\to\infty} \hat{\theta}_i[k] = \frac{1}{n}\sum_{i=1}^{n} \hat{\theta}_i[0] = \frac{1}{n}\sum_{i=1}^{n} H_i^\top z_i$$

- Thus, each sensor asymptotically computes the centralized linear least squares estimate according to

$$\hat{\theta} = \lim_{k\to\infty} P_i[k]^{-1}\,\hat{\theta}_i[k]$$

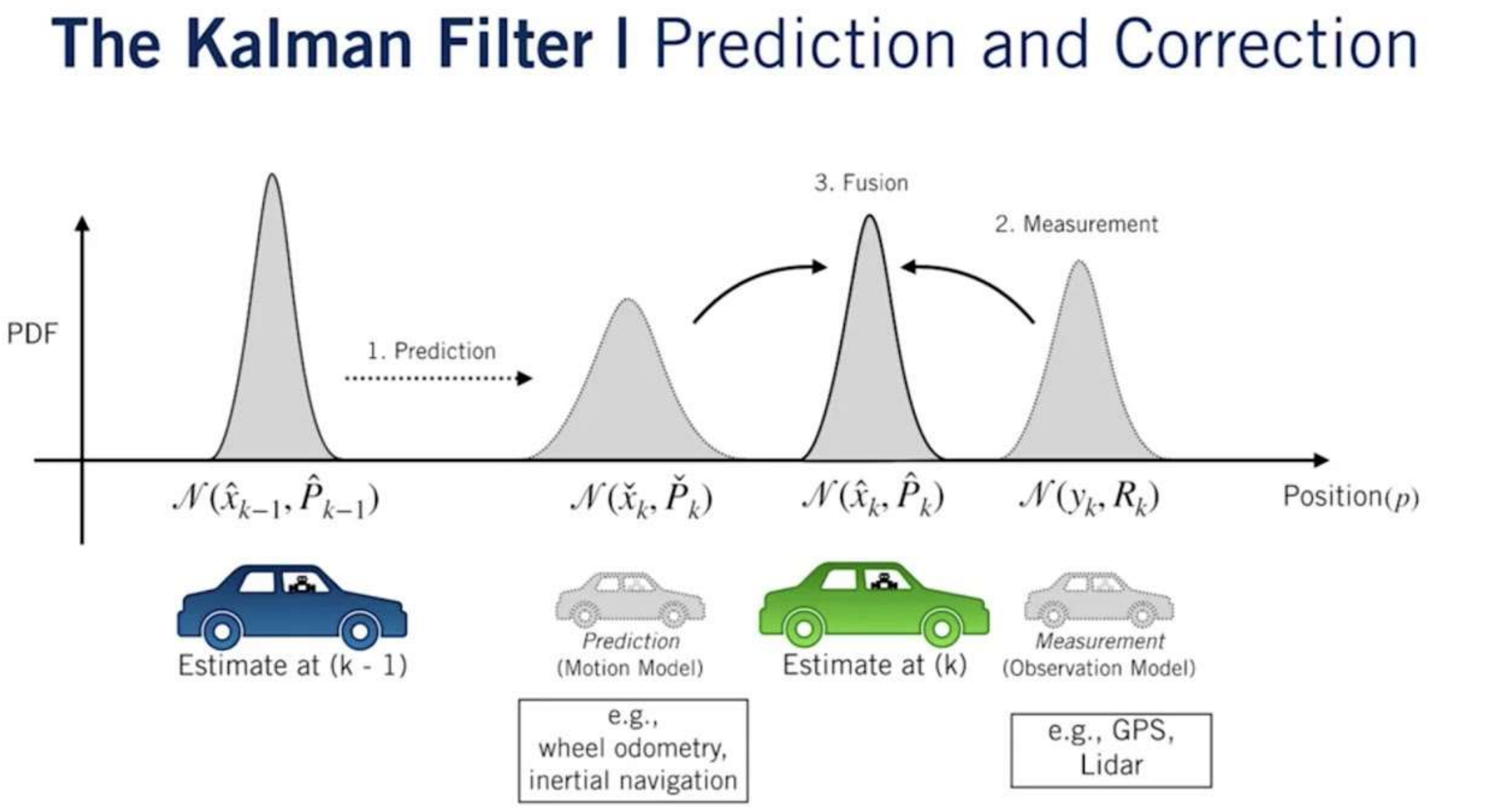
EE185523

Distributed Estimation

Distributed Kalman filter

Yurid E. Nugraha
TSP DTE FTEIC ITS

Kalman filter



Kalman filter

- Variable of interest: state of a linear dynamic system

$$x[k+1] = Ax[k] + w[k] \longrightarrow \text{noise}$$
$$z[k] = H[k]x[k] + v[k] \longrightarrow \text{noise}$$

- Filtering algorithm: observes $z[k]$ and estimate $x[k]$

- Update rule: $\underbrace{\hat{x}[k|k]}_{\text{posterior estimate}} = \hat{x}[k|k-1] + K[k](z[k] - H\underbrace{\hat{x}[k|k-1]}_{\text{prior estimate}})$

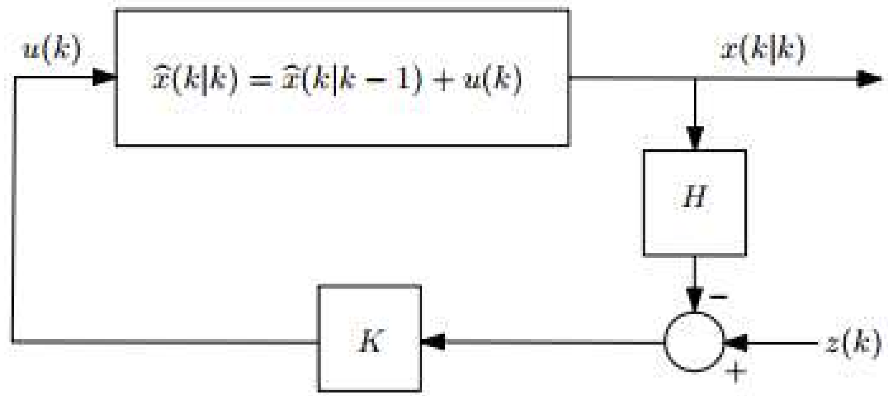
Kalman filter

- Update rule: $\hat{x}[k|k] = \hat{x}[k|k-1] + K[k](z[k] - H\hat{x}[k|k-1])$

  Kalman gain

- K chosen as solution to
$$\min_{K[k]} trace\, \Sigma[k|k]$$
where $\Sigma[k|k] = \mathbb{E}\{\tilde{x}[k|k]\tilde{x}[k|k]^\top\}$ is the covariance matrix of the error vector
$$\tilde{x}[k|k] = \hat{x}[k|k] - x[k]$$

- $K[k] = \Sigma[k|k]H[k]^\top R^{-1}$



Kalman filter

Alternative representation ('information filter'):

- Information matrix $\mathcal{I}[k] = \Sigma[k]^{-1}$
- Recursion:

$$\mathcal{I}[k|k] = \mathcal{I}\;[k|k-1] + Y[k]$$
$$\hat{y}[k|k] = \hat{y}[k|k-1] + y[k]$$

where

$$Y[k] = H[k]^\top V^{-1} H[k]$$
$$y[k] = H[k]^\top V^{-1} z[k]$$

- Then, Kalman gain:

$$K[k] = A\mathcal{I}[k|k]H[k]^\top V^{-1}$$

Meas. noise matrix

Kalman filtering over sensor networks: centralized

- Discrete time system $x[k+1] = Ax[k] + w[k]$
- Observed by n sensors $z_i[k] = H_i[k]x[k] + v_i[k]$
- Fusion center: $z[k] = [z_1[k]^\top, z_2[k]^\top, \dots, z_n[k]^\top]$
- Centralized Kalman filter:

$$x[k+1] = Ax[k] + w[k]$$
$$z[k] = H[k]x[k] + v[k]$$

where $H = \left[H_1[k]; H_2[k]; \dots; H_n[k]\right]^\top$ and $v = \left[v_1[k]; v_2[k]; \dots; v_n[k]\right]^\top$

Kalman filtering over sensor networks: centralized

- Fusion center: $z[k] = [z_1[k]^\top, z_2[k]^\top, \dots, z_n[k]^\top]$
- Centralized Kalman filter:

$$x[k+1] = Ax[k] + w[k]$$
$$z[k] = H[k]x[k] + v[k]$$

where $H = \left[H_1[k]; H_2[k]; \dots; H_n[k]\right]^\top$ and $v = \left[v_1[k]; v_2[k]; \dots; v_n[k]\right]^\top$

- Disadvantage: all the computational work is performed at the fusion center
- Sensors are only employed for gathering the measurements and relaying them to the center.

Kalman filtering over sensor networks: distributed

Additive property for information matrix:

$$\mathcal{I}[k|k] = \mathcal{I}[k|k-1] + \sum_{i=1}^{n} Y_i[k]$$

$$\hat{y}[k|k] = \hat{y}[k|k-1] + \sum_{i=1}^{n} y_i[k]$$

where $Y_i[k] = H_i[k]^\top V_i^{-1} H_i[k]$ and $y_i[k] = H_i[k]^\top V_i^{-1} z_i[k]$

Kalman filtering over sensor networks: distributed

Distributed approach:

- Letting each sensor keep a local copy of the information matrix $\mathcal{I}[k|k-1]$ and the information vector $\hat{y}[k|k-1]$
- When each sensor performs a local Kalman filter based on measurement:

$$H_i[k]^T\, V_i^{-1} H_i[k] \ = \mathcal{I}_i[k|k] - \mathcal{I}_i[k|k-1],$$

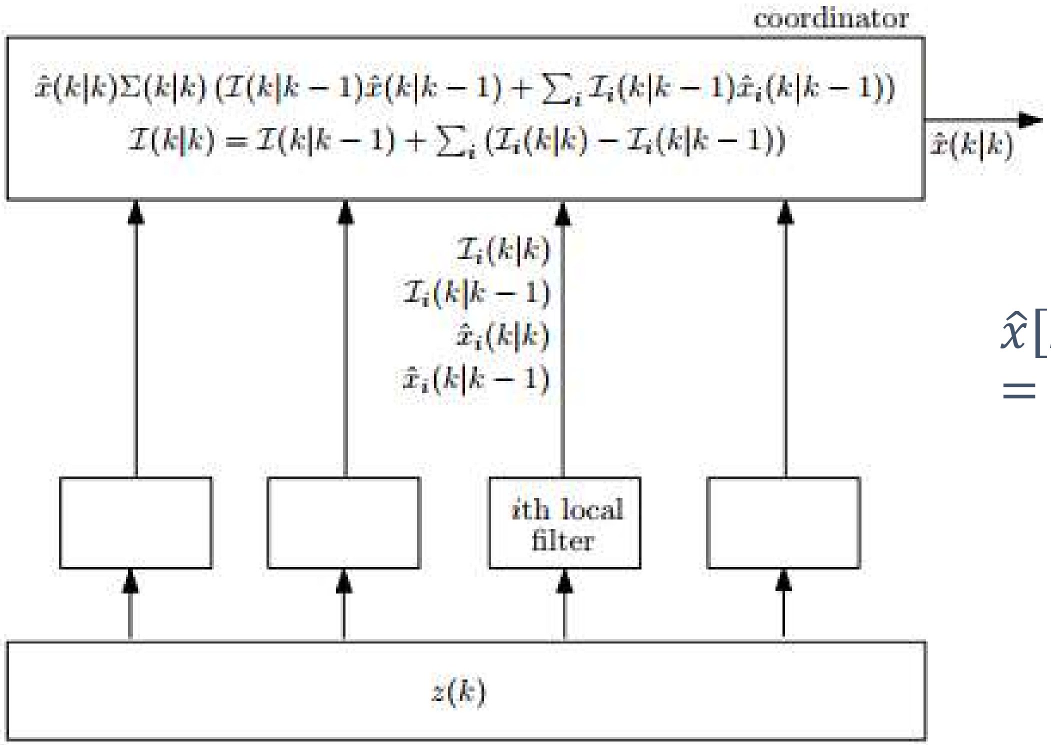
Kalman filtering over sensor networks: distributed

Distributed approach:

- Information matrix can be updated at each node by 1) receiving the difference $\mathcal{J}_i[k|k] - \mathcal{J}_i[k|k-1]$, 2) summing them up across all sensors, and then 3) adding them to obtain $\mathcal{J}_i[k|k-1]$.
- Information vector can be updated by summing up the received $y_i(k)$ from each sensor, which is also the difference $\hat{y}_i[k|k] - \hat{y}_i[k|k-1]$
- The prediction step can now be executed at each node in its original form or in the information filter form

Kalman filtering over sensor networks: distributed

w/ coordinator:

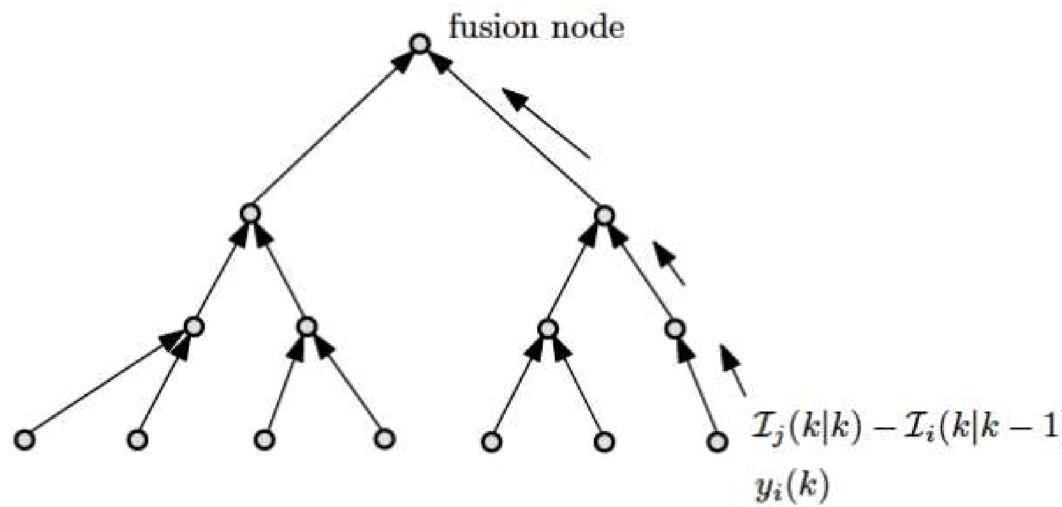


$$\hat{x}[k|k] = \hat{x}[k|k-1] + K(z[k] - H[k]\,\hat{x}[k|k-1])$$

Communication from sensors to coordinator but not vice versa

Relaxing the communication requirement

- Implicitly stated that complete graph is needed
- In-branching graph may relax the requirement



Other approach: combine update equation w/ consensus:

$$\hat{x}_i[k|k] = \hat{x}_i[k|k-1] + K_i^o(z_i[k] - H_i[k]\hat{x}_i[k|k-1]) + \sum_{j\in N_i} K_{ij}^c(\hat{x}_j[k|k-1] - \hat{x}_i[k|k-1])$$

Relaxing the communication requirement

Another distributed approach: Combine update equation for local Kalman filters w/ 'consensus' weight:

$$\hat{x}_i[k|k] = \hat{x}_i[k|k-1] + K_i^o(z_i[k] - H_i[k]\hat{x}_i[k|k-1]) + \sum_{j\in N_i} K_{ij}^c(\hat{x}_j[k|k-1] - \hat{x}_i[k|k-1])$$

- K_i^o: gain by sensor i obtained from its observation
- K_{ij}^c: gain used by sensor i obtained from its communication with j

EE185523

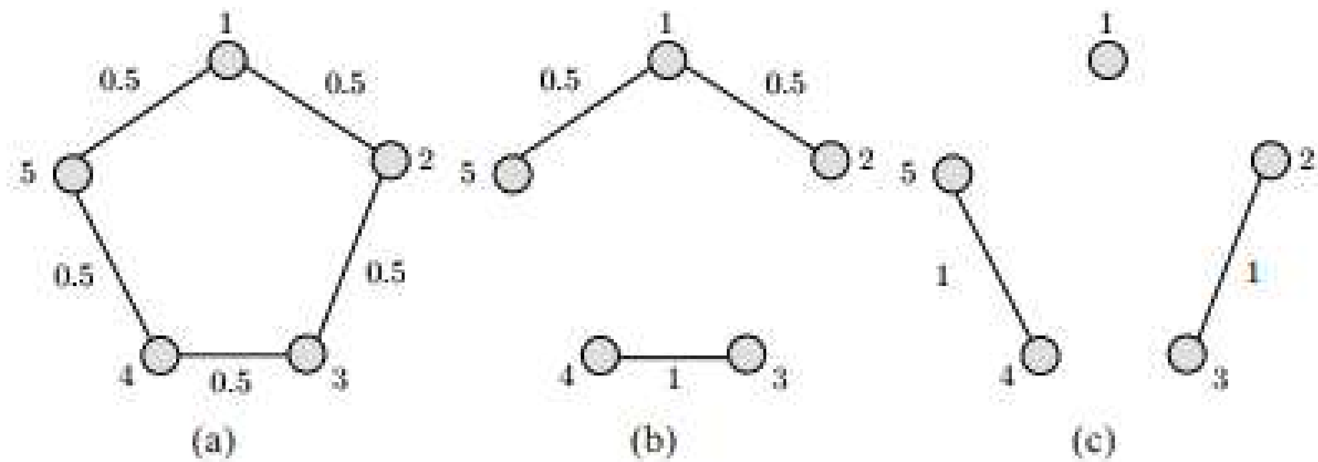
Distributed Estimation

Pulsed intercluster communication

Yurid E. Nugraha
TSP DTE FTEIC ITS

Distributed least squares estimation

- Monolithic vs clustered networks



Monolithic network

Clustered networks w/ different update times

Pulsed intercluster communication

Assumption:

- All **intra**cluster updates occur at every time step $t = k\Delta$
- All **inter**cluster updates occur at every time step $t = k\beta\Delta$
- The set of time instants $[k\beta\Delta + \Delta, k\beta\Delta + 2\Delta, \ldots, (k+1)\beta\Delta]$ constitutes an update cycle
- Evolution of update cycle

$$\hat{\theta}^c\big((k\beta + 1)\Delta\big) = \big(M_w(G_1) - M_w(G_2)\big)\hat{\theta}^c(k\beta\Delta),$$
$$\hat{\theta}^c((k\beta + 2)\Delta) = M_w(G_1)\hat{\theta}^c((k\beta + 1)\Delta) - M_w(G_2)\hat{\theta}^c(k\beta\Delta),$$
$$\cdots$$
$$\hat{\theta}^c((k + 1)\beta\Delta) = M_w(G_1)\hat{\theta}^c(((k + 1)\beta - 1)\Delta) - M_w(G_2)\hat{\theta}^c(k\beta\Delta)$$

Clustered networks: vector case

The total number of edges is 163, of which 11 correspond to the intercluster network (dotted lines).