# Chapter 1

# Single-Target, Single-Sensor Filtering

## 1.1 The Bayesian Approach

In the general Bayesian filtering framework for systems with Gaussian process and measurement noise, we consider the following system:

$$\mathbf{x}_k = f(\mathbf{x}_{k-1}, \mathbf{w}_{k-1}) \tag{1.1a}$$

$$\mathbf{y}_k = h(\mathbf{x}_{k-1}, \mathbf{v}_{k-1}) \tag{1.1b}$$

where $\mathbf{x}_k \in \mathbb{R}^n, \mathbf{y}_k \in \mathbb{R}^m, m \leq n, f : \mathbb{R}^n \to \mathbb{R}^n, h : \mathbb{R}^n \to \mathbb{R}^m$ are possibly nonlinear, time-varying functions, and $w_k$ are i.i.d, $v_k$ are i.i.d.

We will further assume a *Markovian* setting: 1) given $\mathbf{x}_k$, $\mathbf{x}_{k+1}$ is independent of all past states $\mathbf{x}_0, \cdots, \mathbf{x}_{k-1}$ and all past observations $\mathbf{y}_1, \cdots, \mathbf{y}_k$, and 2) given $\mathbf{x}_k$, $\mathbf{y}_k$ is independent of the past states and the past measurements.

The objective of a *filtering problem* is to estimate the full state $\mathbf{x}_k$ given data observations $\mathscr{A}_k := \sigma(\mathbf{y}_1, \cdots, \mathbf{y}_k)$. Taking a Bayesian approach, we construct the pdf $p(\mathbf{x}_k|\mathscr{A}_k)$ given $p(\mathbf{x}_{k-1}|\mathscr{A}_{k-1})$ at each timestep $k$ through a two-step process: prediction nd measurement update.

1. **Prediction**: given $p(\mathbf{x}_{k-1}|\mathscr{A}_{k-1})$, use the Chapman-Kolmogorov equation to predict $\mathbf{x}_k$

   $$p(\mathbf{x}_k|\mathscr{A}_{k-1}) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathscr{A}_{k-1})p(\mathbf{x}_{k-1}|\mathscr{A}_{k-1})d\mathbf{x}_{k-1} = \int p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-1}|\mathscr{A}_{k-1})d\mathbf{x}_{k-1} \tag{1.2}$$

   where the second equality follows from Markovian assumptions. In this form, $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ can be directly determined from the system dynamics.

2. **Measurement Update**: given $p(\mathbf{x}_k|\mathscr{A}_{k-1})$, use Bayes' Rule to incorporate new data point $\mathbf{y}_k$ into our prediction:

   $$p(\mathbf{x}_k|\mathscr{A}_k) = \frac{p(\mathbf{x}_k, \mathbf{y}_k|\mathscr{A}_{k-1})}{p(\mathbf{y}_k|\mathscr{A}_{k-1})} = \frac{p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathscr{A}_{k-1})}{p(\mathbf{y}_k|\mathscr{A}_{k-1})} \tag{1.3}$$

   where the second equality follows from Markovian assumptions, $p(\mathbf{y}_k|\mathbf{x}_k)$ can be determined from the observation equation, and

   $$p(\mathbf{y}_k|\mathscr{A}_{k-1}) = \int p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathscr{A}_{k-1})d\mathbf{x}_k$$

The *prediction problem*, which predicts $\mathbf{x}_{k+1}$ from past data observations $\mathscr{A}_k := \sigma(\mathbf{y}_1, \cdots, \mathbf{y}_k)$, is very similar to the filtering problem: simply reverse the above two steps to construct the pdf $p(\mathbf{x}_{k+1}|\mathscr{A}_k)$ given $p(\mathbf{x}_k|\mathscr{A}_{k-1})$.

Consider the following general nonlinear system

$$d\mathbf{x}(t) = F_c(t, \mathbf{x}(t), \mathbf{w}(t))dt \tag{1.4a}$$
$$\mathbf{y}(t) = h(\mathbf{x}(t), \mathbf{v}(t)) \tag{1.4b}$$

where $\mathbf{x}(t) \in \mathcal{X} := \mathbb{R}^{n_x}, \mathbf{y}(t) \in \mathcal{Y} := \mathbb{R}^{n_y}, n_y \leq n_x$, $F_c : \mathbb{R}^+ \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_w} \to \mathbb{R}^{n_x}, h : \mathbb{R}^{n_x} \times \mathbb{R}^{n_v} \to \mathbb{R}^{n_y}$ are possibly nonlinear, time-varying functions, stochastic disturbances $\mathbf{w}(t)$ are i.i.d, $\mathbf{v}(t)$ are i.i.d and belong on probability space $(\Omega, \mathcal{F}, P)$, where $\Omega$ denotes the sample space of all possible outcomes, $\mathcal{F}$ is the set of events, and $P$ is the probability function.

We assume that the system is *Markovian*: 1) given $\mathbf{x}(s)$ for some $s \leq t$, $\mathbf{x}(t)$ is independent of $\{\mathbf{x}(r) : r < s\}$ and all past measurements $\{\mathbf{y}(r) : r < s\}$, 2) given $\mathbf{x}(t)$, $\mathbf{y}(t)$ is independent of all past states and measurements.

For state estimation problems, it is often convenient to discretize the system with some (possibly fixed) sample time $\Delta t > 0$.

$$\mathbf{x}_{k+1} = F_d(k, \mathbf{x}_k, \mathbf{w}_k) \tag{1.5a}$$
$$\mathbf{y}_k = h(\mathbf{x}_k, \mathbf{v}_k) \tag{1.5b}$$

This discretized system is also assumed to be Markovian in a similar way as its continuous-time analog. An alternative way of describing (1.5) is through the use of probability distributions. Denote $p(\mathbf{x}_{k+1}|\mathbf{x}_k)$ to be the probability of transitioning from $\mathbf{x}_k$ to $\mathbf{x}_{k+1}$ for any pair of states $\mathbf{x}_k, \mathbf{x}_{k+1} \in \mathcal{X}$. Further denote $\ell(\mathbf{y}_k|\mathbf{x}_k)$ to be the likelihood probability of observing $\mathbf{y}_k \in \mathcal{Y}$ given $\mathbf{x}_k \in \mathcal{X}$.

The objective of the *filtering problem* is to estimate the full state $\mathbf{x}_k$ given data observations $\mathscr{A}_k := \sigma(\mathbf{y}_1, \cdots, \mathbf{y}_k)$. Taking a Bayesian approach, we construct the pdf $f(\mathbf{x}_k|\mathscr{A}_k)$ given $f(\mathbf{x}_{k-1}|\mathscr{A}_{k-1})$ at each timestep $k$ through a two-step process: prediction and measurement update.

1. **Prediction**: given $f(\mathbf{x}_{k-1}|\mathscr{A}_{k-1})$, use the Chapman-Kolmogorov equation to predict $\mathbf{x}_k$

$$f(\mathbf{x}_k|\mathscr{A}_{k-1}) = \int f(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathscr{A}_{k-1}) f(\mathbf{x}_{k-1}|\mathscr{A}_{k-1}) d\mathbf{x}_{k-1} = \int f(\mathbf{x}_k|\mathbf{x}_{k-1}) f(\mathbf{x}_{k-1}|\mathscr{A}_{k-1}) d\mathbf{x}_{k-1} \tag{1.6}$$

   where the second equality follows from Markovian assumptions. In this form, $f(\mathbf{x}_k|\mathbf{x}_{k-1})$ can be directly determined from the system dynamics.

2. **Measurement Update**: given $f(\mathbf{x}_k|\mathscr{A}_{k-1})$, use Bayes' Rule to incorporate new data point $\mathbf{y}_k$ into our prediction:

$$f(\mathbf{x}_k|\mathscr{A}_k) = \frac{f(\mathbf{x}_k, \mathbf{y}_k|\mathscr{A}_{k-1})}{f(\mathbf{y}_k|\mathscr{A}_{k-1})} = \frac{\ell(\mathbf{y}_k|\mathbf{x}_k) f(\mathbf{x}_k|\mathscr{A}_{k-1})}{f(\mathbf{y}_k|\mathscr{A}_{k-1})} \tag{1.7}$$

   where the second equality follows from Markovian assumptions, $f(\mathbf{y}_k|\mathbf{x}_k)$ can be determined from the observation equation, and

$$f(\mathbf{y}_k|\mathscr{A}_{k-1}) = \int \ell(\mathbf{y}_k|\mathbf{x}_k) f(\mathbf{x}_k|\mathscr{A}_{k-1}) d\mathbf{x}_k$$

**Remark 1.** The two steps are alternatively referred to as the **prior estimate** and the **posterior update**. □

**Remark 2.** The *prediction problem*, which predicts $\mathbf{x}_{k+1}$ from past data observations $\mathscr{A}_k := \sigma(\mathbf{y}_1, \cdots, \mathbf{y}_k)$, is very similar to the filtering problem: simply reverse the above two steps to construct the pdf $f(\mathbf{x}_{k+1}|\mathscr{A}_k)$ given $f(\mathbf{x}_k|\mathscr{A}_{k-1})$. □

One primary issue is that for general nonlinear systems, a direct implementation of the above two steps is often difficult to do. Throughout this chapter, we will be focusing on additive noise perturbations. That is, we consider (1.4) specifically of the form

$$d\mathbf{x}(t) = f_c(t, \mathbf{x}(t))dt + d\mathbf{w}(t) \tag{1.8a}$$

$$\mathbf{y}(t) = h(\mathbf{x}(t)) + \mathbf{v}(t) \tag{1.8b}$$

and the corresponding discrete-time system

$$\mathbf{x}_{k+1} = f_d(k, \mathbf{x}_k) + \mathbf{w}_k \tag{1.9a}$$

$$\mathbf{y}_k = h(\mathbf{x}_k) + \mathbf{v}_k \tag{1.9b}$$

By way of the *Orthogonality Principle*, we can show that the *optimal estimator* $\hat{\mathbf{x}}(t)$ of $\mathbf{x}(t)$ is given by $\hat{\mathbf{x}}(t) = \mathbb{E}_P[\mathbf{x}(t)|\mathscr{A}(t)]$ and likewise, in the discrete-time case, $\hat{\mathbf{x}}_k = \mathbb{E}_P[\mathbf{x}_k|\mathscr{A}_k]$. Hence, we discuss the Orthgonality Principle in the following section.

## 1.2 The Orthogonality Principle

Define $\mathcal{L}^2 := \mathcal{L}^2(\Omega, \mathcal{F}, P)$ to be the space of all random variables on $\Omega \subseteq \mathbb{R}^n$ with finite second moments, i.e, the space of $\mathbf{x}$ such that $\mathbb{E}_P[\|\mathbf{x}\|^2] < \infty$. To maintain simplicity in this section, we use the notation $\mathbf{x}$, without mentioning the time index, to denote the random variable $\mathbf{x}(t)$ or $\mathbf{x}_k$ instead of the actual time-varying trajectory. Let $\mathcal{V}$ be a closed, linear subspace on $\mathcal{L}^2$.

With this setup, the estimation problem that the Bayesian filtering process posed to us in the previous section can be restated as follows: determine the estimator of $\mathbf{x} \in \mathcal{L}^2$ over all $\mathcal{V}$ with the least mean-squared error, i.e., we want to find a $\mathbf{z}^* \in \mathcal{V}$ such that $\mathbb{E}_P[\|\mathbf{x} - \mathbf{z}^*\|^2] \leq \mathbb{E}_P[\|\mathbf{x} - \mathbf{z}\|^2]$ for all $z \in \mathcal{V}$. Correspondingly, $\mathbb{E}_P[\|\mathbf{x} - \mathbf{z}^*\|^2]$ is known as the *minimum mean squared error (MMSE)*. See Figure 1.1 for visualization in the case where $\mathcal{V}$ is a line.

**Theorem 1** (Orthogonality Principle)**.** The following conditions are equivalent:

1. There exists a **unique** element $\mathbf{z}^* \in \mathcal{V}$ which achieves the MMSE.
2. Let $\mathbf{y} \in \mathcal{L}^2$. Then $\mathbf{y} = \mathbf{z}^*$ iff the following two conditions hold:
   a. $\mathbf{y} \in \mathcal{V}$
   b. $(\mathbf{x} - \mathbf{y}) \perp \mathbf{z}$ for all $\mathbf{z} \in \mathcal{V}$.
3. As a consequence of the above two conditions, the MMSE has a nice simplification:

$$\mathbb{E}_P[(\mathbf{x} - \mathbf{z}^*)^T(\mathbf{x} - \mathbf{z}^*)] = \mathbb{E}_P[\mathbf{x}^T\mathbf{x}] - \mathbb{E}_P[(\mathbf{z}^*)^T\mathbf{z}^*] \quad \text{since } (\mathbf{x} - \mathbf{z}^*) \perp \mathbf{z}^*$$

Furthermore, the MMSE is an *unbiased estimator*, meaning $\mathbb{E}_P[\mathbf{x} - \mathbf{z}^*] = 0$. $\mathbf{z}^*$ is alternatively known as the *projection* of $x$ onto $\mathcal{V}$.

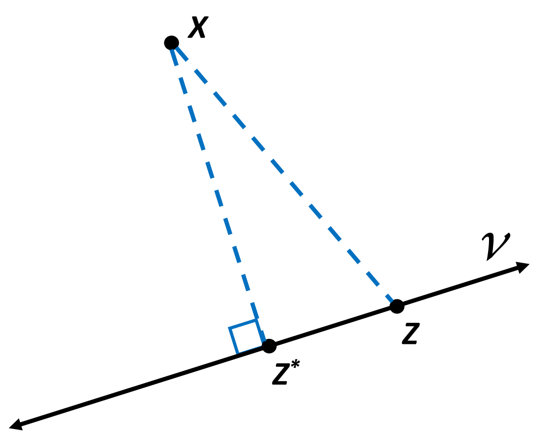

**Figure 1.1:** A visualization of the projection mechanism where $\mathcal{V}$ is a line and $\mathbf{x} \in \mathbb{R}^n$. Note that any other $\mathbf{z} \in \mathcal{V}$ does not achieve the minimum error.

**Example 1** (Specialization to Linear Subspaces)**.** One of the most common classes of estimators to constrain $\mathbf{z}^*$ to are linear estimators, which consequently gives rise to the *Kalman filtering* algorithm and its variants. We specialize the above analysis to the affine subspace $\mathcal{V} = \{c_0 + c_1 y_1 + \cdots + c_m y_m; c_i \in \mathbb{R}^n\}$ for a specific value $\mathbf{y} := (y_1, \cdots, y_m)$. The projection of $\mathbf{x}$ onto $\mathcal{V}$ is of the form $\mathbf{z}^* = A\mathbf{y} + \mathbf{b}$. The estimation error becomes $\mathbf{e} = \mathbf{x} - (A\mathbf{y} + \mathbf{b})$. Use the Orthogonality Principle to determine what the coefficients $A$ and $\mathbf{b}$ should be. Namely, in order to have $\mathbf{e} \perp \mathbf{z}$ for all $\mathbf{z} \in \mathcal{V}$, we need:

- $\mathbb{E}_P[\mathbf{e}] = 0$, which implies that $\mathbf{b} = \mathbb{E}_P[\mathbf{x}] - A\mathbb{E}_P[\mathbf{y}]$. Thus, $z^* = \mathbb{E}_P[\mathbf{x}] + A(\mathbf{y} - \mathbb{E}_P[\mathbf{y}])$.
- $\text{Cov}(\mathbf{e}, \mathbf{y}) = 0$. Combined with the previous expression, we get:

$$\begin{aligned} \text{Cov}(\mathbf{x} - \mathbb{E}_P[\mathbf{x}] - A(\mathbf{y} - \mathbb{E}_P[\mathbf{y}]), \mathbf{y}) = 0 &\implies \text{Cov}(\mathbf{x}, \mathbf{y}) - A\text{Cov}(\mathbf{y}) = 0 \\ &\implies A = \text{Cov}(\mathbf{x}, \mathbf{y})\text{Cov}(\mathbf{y})^{-1} \end{aligned}$$

Combined together, we have the final expression for the MMSE. To distinguish the notation between the general and the linear cases, we denote the conditional expectation for the linear case with $\hat{E}$ instead of the usual $\mathbb{E}$.

$$\hat{\mathbf{x}} := \hat{E}[\mathbf{x}|\mathbf{y}] := \mathbb{E}_P[\mathbf{x}] + \text{Cov}(\mathbf{x}, \mathbf{y})\text{Cov}(\mathbf{y})^{-1}(\mathbf{y} - \mathbb{E}_P[\mathbf{y}]) \tag{1.10}$$

Moreover, we find that $\text{Cov}(\mathbf{e})$ satisfies:

$$\text{Cov}(\mathbf{e}) = \text{Cov}(\mathbf{x}) - \text{Cov}(\hat{\mathbf{x}}) = \text{Cov}(\mathbf{x}) - \text{Cov}(\mathbf{x}, \mathbf{y})\text{Cov}(\mathbf{y})^{-1}\text{Cov}(\mathbf{y}, \mathbf{x})$$

□

## 1.3 The Particle Filter

Now we derive optimal estimators for the continuous-time system (1.8) in the case where $d\mathbf{w}(t) = \sigma(t)dW(t)$ for the standard Brownian motion process $W(t)$. When the dynamics are possibly nonlinear, it is common to use two kinds of filtering techniques: 1) particle filtering, or 2) linearizing the dynamics to take advantage of Gaussian properties. We begin by discussing particle filters in this section. It turns out that we can invoke measure transformation procedures from stochastic process theory to aid with the design of the particle filter. The formal background on change of measures, including the well-known Girsanov's formula, is presented first.

### 1.3.1 Change of Measure for Gaussian Distributions

**Definition 1** (Absolute Continuity)**.** A measure $Q$ is said to be *absolutely continuous* with respect to another measure $P$ (also denoted mathematically by $Q << P$) if the null set of $P$ is also under the null set of $Q$. That is, for any event set $E$, if $P(E) = 0$ then $Q(E) = 0$. □

**Definition 2** (Radon-Nikodym Derivative)**.** The Radon-Nikodym derivative is defined as the multiplicative factor which transforms from measure $P$ to measure $Q$, for $Q << P$:

$$Z(x) = \frac{dQ(x)}{dP(x)} \implies dQ(x) = Q(X = x)dx = Z(x)dP(x) \tag{1.11}$$

In terms of conditional expectations:

$$\mathbb{E}_P[Z(x)] = \int Z(x)dP(x) = \int dQ(x) = \mathbb{E}_Q[1]$$

and for general functions $h \in \mathcal{L}^1$, $\mathbb{E}_P[Z(x)h(x)] = \mathbb{E}_Q[h(x)]$.

In the case of the scalar Gaussian random variable, the interpretation of the Radon-Nikodym derivative goes as follows: with respect to measure $P$, $X$ is normally distributed with mean $\mu$, variance $\sigma^2$, and with respect to measure $Q$, $X$ is normally distributed with mean 0 (same variance $\sigma^2$). □

There are two analogous change-of-measure formulas for the Gaussian stochastic random process:

1. Discrete-Time Case: We begin with $X_n = \sum_{i=1}^{n} \Delta X_i \sim \mathcal{N}[0, t_n]$ under measure $P$. The shift term is given by

   $$Y_n = \sum_{i=1}^{n} \mu_i \Delta t$$

   which corresponds to the Radon-Nikodym derivative

   $$Z_n = \frac{dQ}{dP} = e^{\sum\limits_{i=1}^{n} \mu_i \Delta x_i - \frac{1}{2} \sum\limits_{i=1}^{n} \mu_i^2 \Delta t} \tag{1.12}$$

   This causes the shift to $X_n - \sum_{i=1}^{n} \mu_i \Delta t \sim \mathcal{N}(0, t_n)$ under new measure $Q$.

2. Continuous-Time Case: We begin with $W(t)$, the standard Brownian motion process, under measure $P$. The shift term is given by

   $$Y(t) = \int_0^t \mu(s)ds$$

   which corresponds to the Radon-Nikodym derivative

   $$Z(t) = \frac{dQ}{dP} = e^{\int_0^t \mu(s)dW(s) - \frac{1}{2}\int_0^t \mu^2(s)ds} \tag{1.13}$$

This causes the shift to $\tilde{W}(t) := W(t) - \int_0^t \mu(s)ds$ under new measure $Q$. Furthermore, the Radon-Nikodym derivative $Z(t)$ for this Brownian motion case is commonly referred to as the **Doléans-Dade exponential**. A further treatment of the continuous-time case can be found in [1].

**Proposition 1.** The Doléans-Dade exponential $Z(t)$ (1.13) is a martingale.

In summary, (1.13) is the solution to the SDE $dZ(t) = Z(t)\mu(t)dW(t)$, and encompasses the transformation $\tilde{W}(t) = W(t) - \mu(t)$. One can additionally extend the Doléans-Dade exponential to the vector case. Suppose $W(t) \in \mathbb{R}^d$ and $\overrightarrow{\mu}(t) \in \mathbb{R}^d$ is the desired shift in mean. Then (1.13) in the vector case is written as:

$$Z(t) = e^{\int_0^t \overrightarrow{\mu}^T(s)dW(s) - \frac{1}{2}\int_0^t \overrightarrow{\mu}^T(s)\overrightarrow{\mu}(s)ds} \tag{1.14}$$

and is the solution to the SDE $dZ(t) = Z(t)\mu^T(t)dW(t)$. Note that the Radon-Nikodym derivative $Z(t)$ is still a scalar quantity.

**Lemma 1** (Bayes' Theorem for Conditional Expectations)**.** Let $(\Omega, \mathcal{F}, P)$ be a probability space and $Q$ be such that $Q << P$. Let the multiplicative change of measure term be denoted by $Z$ denote the Radon-Nikodym derivative transforming from $P$ to $Q$. Then for any $\sigma$-algebra $\mathcal{G} \subset \mathcal{F}$, and any integrable random variable $X \in \mathbb{R}$ such that $\mathbb{E}_Q[|X|] < \infty$, the Bayes' formula holds:

$$\mathbb{E}_Q[X|\mathcal{G}] = \frac{\mathbb{E}_P[ZX|\mathcal{G}]}{\mathbb{E}_P[Z|\mathcal{G}]}$$

In the case of random vectors, $\mathbf{x} \in \mathbb{R}^n$ such that $\mathbb{E}_Q[\|\mathbf{x}\|] < \infty$ where the norm can be any norm, Bayes' formula is:

$$\mathbb{E}_Q[\mathbf{x}|\mathcal{G}] = \frac{\mathbb{E}_P[Z\mathbf{x}|\mathcal{G}]}{\mathbb{E}_P[Z|\mathcal{G}]}$$

Note that the Radon-Nikodym derivative is still a scalar.

We can invoke **Lévy's Characterization of the Brownian motion** to show that the transformed white noise process is indeed a Brownian motion under the new measure $Q$, i.e., the new $\tilde{W}(t) := W(t) - \mu t$ is a martingale. The characterization consists of three parts:

1. $P(\tilde{W}(0) = 0) = 1$
2. $\tilde{W}(t)$ is a martingale with continuous sample paths
3. $\tilde{W}^2(t) - t$ is a martingale.

The proofs of this characterization and Bayes' rule can be found in multiple texts, so we do not discuss them here.

Now that we have discussed the change of measure between Brownian motion processes, we discuss how such a change of measure affects SDEs perturbed by Brownian motion processes. In particular, changing between two drift terms of the additive-noise SDE will be important for our discussion of measure transformations in the particle filter.

**Example 2** (Drift-to-Drift Transformation)**.** Consider the following example transformations of the Brownian motion process. Let $\sigma(t) \not\equiv 0$ for all $t$.

1. Suppose we want to transform a drift-perturbed Brownian motion process to a process without drift.

$$dX(t) = \mu(t)dt + \sigma(t)dW(t) = \sigma(t)\left(\frac{\mu}{\sigma(t)}dt + dW(t)\right)$$

   We want to find a $\tilde{W}(t)$ such that $dX(t) = \sigma(t)d\tilde{W}(t)$.
   Choose the shift term $Y(t) = -(\mu(t)/\sigma(t))t$ so that $\tilde{W}(t) = W(t) + (\mu(t)/\sigma(t))t$. Then:

$$\tilde{Z}(t) = \frac{dQ(t)}{dP(t)} = e^{-\frac{\mu(t)}{\sigma(t)}W(t) - \frac{1}{2}\left(\frac{\mu(t)}{\sigma(t)}\right)^2 t}$$

2. Now suppose we want to transform a zero-drift Brownian motion process to one with nonzero drift.

$$dX(t) = \sigma(t)\tilde{W}(t) = \nu(t)dt + (-\nu(t)dt + \sigma(t)\tilde{W}(t)) = \nu(t)dt + \sigma(t)\left(-\frac{\nu(t)}{\sigma(t)}dt + \tilde{W}(t)\right)$$

   We want to find $W'(t)$ such that $dX(t) = \nu(t)dt + \sigma(t)dW'(t)$.
   Choose the shift term $\tilde{Y}(t) = (\nu(t)/\sigma(t))t$ so that $W'(t) = \tilde{W}(t) - (\nu/\sigma)t$. Then:

$$Z'(t) = \frac{dR(t)}{dQ(t)} = e^{\frac{\nu(t)}{\sigma(t)}\tilde{W}(t) - \frac{1}{2}\left(\frac{\nu(t)}{\sigma(t)}\right)^2 t}$$

Overall, the change of measure from drift $\mu(t)$ to drift $\nu(t)$ is performed using the Radon-Nikodym derivative

$$\frac{dR(t)}{dP(t)} = \frac{\cancel{dQ(t)}}{dP(t)} \cdot \frac{dR(t)}{\cancel{dQ(t)}} = \tilde{Z}(t)Z'(t)$$

with

$$dW'(t) = dW(t) - \frac{\nu(t) - \mu(t)}{\sigma(t)}dt$$

$\square$

### 1.3.2 Change of Measure Related to Particle Filtering

For the dynamics (1.8) with $d\mathbf{w}(t) = \sigma(t)dW(t)$, suppose we additionally keep track of an *importance process* of the form

$$d\mathbf{z}(t) = g(t, \mathbf{z}(t))dt + \theta(t)dW(t) \tag{1.15}$$

where the Brownian motion process $W(t)$ is the same as that of the state dynamics.

**Theorem 2** (Transformation of Solutions via Change of Measure for Gaussian Noise Processes)**.** The process

$$d\mathbf{z}^*(t) = \sigma(t)\theta(t)^{-1}d\mathbf{z}(t), \quad \mathbf{z}(0) = \mathbf{x}_0 \tag{1.16}$$

yields a weak solution to (1.8) under the measure $Q$, where $Z(t)$ is the Radon-Nikodym derivative of the form

$$Z(t) = e^{\int_0^t \vec{\mu}(s)^T dW(s) + \frac{1}{2}\int_0^t \vec{\mu}(s)^T \vec{\mu}(s)ds}$$

*Proof.* Substitute (1.15) into (1.16) to get

$$d\mathbf{z}^*(t) = \sigma(t)\theta^{-1}(t)g(t,\mathbf{z}(t))dt + \sigma(t)dW(t)$$

Then solving for $dW(t)$ yields:

$$dW(t) = \sigma(t)^{-1}d\mathbf{z}^*(t) - \theta^{-1}(t)g(t,\mathbf{z}(t))dt \tag{1.17}$$

Define

$$\overrightarrow{\mu}(t) := \sigma(t)^{-1}f_c(t,\mathbf{z}^*(t)) - \theta(t)^{-1}g(t,\mathbf{z}(t)) \tag{1.18}$$

To provide intuition behind why we create such a definition, we take note of its analogy to the simple 1D case described in Example 2. The drift (mean) of the original importance process is $g(t,\mathbf{z}(t))$, but it needs to be transformed to the new mean $f_c(t,\mathbf{z}^*(t))$.

Then we can define the new process as follows

$$d\tilde{W}(t) = dW(t) - \overrightarrow{\mu}(t)dt$$

and we can derive the corresponding form of the Doléans-Dade exponential. Under the transformed measure $Q$, this noise process is a Brownian motion process. Simplifying the expression by substituting in $\overrightarrow{\mu}(t)$ and (1.17):

$$\begin{aligned} d\tilde{W}(t) &= dW(t) - \sigma(t)^{-1}f_c(t,\mathbf{z}^*(t))dt + \theta(t)^{-1}g(t,\mathbf{z}(t))dt \\ &= \sigma(t)^{-1}d\mathbf{z}^*(t) - \sigma(t)^{-1}f_c(t,\mathbf{z}^*(t))dt \end{aligned}$$

and rearranging the equation yields the transformed SDE:

$$d\mathbf{z}^*(t) = f_c(t,\mathbf{z}^*(t))dt + \sigma(t)d\tilde{W}(t)$$

This tells us that $\mathbf{z}^*(t)$ is the solution to (1.8) under the new measure $Q$. ■

**Example 3** (Linear Case)**.** We can adapt this analysis specially to the case of linear dynamics:

$$\begin{aligned} d\mathbf{x}(t) &= A\mathbf{x}(t)dt + \sigma(t)dW(t), \quad \mathbf{x}(0) = \mathbf{x}_0 \\ \mathbf{y}(t) &= C\mathbf{x}(t) + \mathbf{v}(t) \end{aligned} \tag{1.19}$$

We take the importance process to be

$$d\mathbf{z}(t) = B\mathbf{z}(t)dt + \theta(t)dW(t)$$

and consider again the transformation

$$d\mathbf{z}^*(t) = \sigma(t)\theta^{-1}(t)d\mathbf{z}(t) = \sigma(t)\theta^{-1}(t)B\mathbf{z}(t)dt + \sigma(t)dW(t)$$

Define again the difference of the two drift terms:

$$\overrightarrow{\mu}(t) = \sigma^{-1}(t)A\mathbf{z}^*(t) - \theta^{-1}(t)B\mathbf{z}(t)$$

Then the new Brownian motion process can be constructed as

$$d\tilde{W}(t) = dW(t) - \overrightarrow{\mu}(t)dt = \sigma^{-1}(t)\left(d\mathbf{z}^*(t) - A\mathbf{z}^*(t)dt\right)$$

and when this is rearranged:

$$d\mathbf{z}^*(t) = A\mathbf{z}^*(t)dt + \sigma(t)d\tilde{W}(t)$$

which implies that $\mathbf{z}^*(t)$ is a solution to (1.19) under the new measure $Q$. The corresponding Doléans-Dade exponential is:

$$Z(t) = \exp\left\{\int_0^t \left[\sigma^{-1}(s)A\mathbf{z}^*(s) - \theta^{-1}(s)B\mathbf{z}(s)\right]^T dW(s) \right.$$
$$\left. -\frac{1}{2}\int_0^t \left[\sigma^{-1}(s)A\mathbf{z}^*(s) - \theta^{-1}(s)B\mathbf{z}(s)\right]^T \left[\sigma^{-1}(s)A\mathbf{z}^*(s) - \theta^{-1}(s)B\mathbf{z}(s)\right]\right\}$$

□

## 1.4 The Kalman Filter

In the previous section, we discussed the relationship of probability measure transformations, a well-known concept in the study of stochastic differential equations, to particle filtering. When the dynamics are linear and perturbed by an additive Gaussian noise process, as in Example 3, two additional properties make the filtering approach more straightforward than the techniques we've seen previously. Namely,

- **Affine combinations of Gaussian random vectors are still Gaussian-distributed.** So, if our initial condition $\mathbf{x}(0)$ has a prior distribution which is Gaussian, all future state variables $\mathbf{x}(t)$ (or $\mathbf{x}_k$, in the discrete-time case) will be Gaussian-distributed too.
- **Gaussian distributions are fully characterized by its mean and covariance matrix.** As a consequence of this property, determining the mean and the covariance of the true state is enough to know everything about the full distribution.

With these two properties, state estimation can be achieved by using a type of transformation different from the measure transformation approach seen in Example 3. It is based off of the construction of *innovation processes*. The primary distinction is that the space of measurements $\mathscr{A}(t)$ is the quantity that is changed instead of changing $\mathbf{x}(t)$ via multiplication of the Radon-Nikodym derivative $Z(t)$. This gives rise to the *Kalman filtering algorithm*, which is the focus of this present section.

### 1.4.1 Linear Innovations Sequence

Before we derive the Kalman filtering process, we first establish some necessary background about innovation processes, particularly in the case where the conditioned space is linear.

The motivation behind the construction of innovation sequences is as follows: it will often be the case that a new observation $\mathbf{y}[t]$ at time $t \in \mathbb{N}$ is not totally new if we've already observed previous values $\mathbf{y}[1], \cdots, \mathbf{y}[t-1]$. The only innovation will come from the component that is orthogonal to the linear span of all the previous observations:

$$\tilde{\mathbf{y}}[t] = \mathbf{y}[t] - \hat{E}[\mathbf{y}[t]|\mathscr{A}[t-1]] \tag{1.20}$$

where $\hat{E}$ is the notation derived from (1.10), and $\mathscr{A}[s]$ represents the sigma algebra $\sigma(\mathbf{y}[1], \cdots, \mathbf{y}[s])$ spanned by the random vectors $\mathbf{y}[1], \cdots, \mathbf{y}[s]$ for any $s \in \mathbb{N}$. Furthermore, with this definition, $\mathbb{E}[\tilde{\mathbf{y}}[t]] = 0$.

For this simplicity, we will often condition our estimate $\hat{\mathbf{x}}$ around this *linear innovations sequence* $\tilde{\mathbf{y}}[1], \tilde{\mathbf{y}}[2], \cdots$ as opposed to the original observation sequence $\mathbf{y}[1], \mathbf{y}[2], \cdots$.

**Theorem 3.** The estimate of $\mathbf{x}$ based on the sequence of orthogonal observations $\tilde{\mathbf{y}}[1], \tilde{\mathbf{y}}[2], \cdots, \tilde{\mathbf{y}}[t]$ is given by the joint projection:

$$\hat{E}[\mathbf{x}|\tilde{\mathscr{A}}[t]] = \overline{\mathbf{x}} + \sum_{i=1}^{t} \hat{E}[\mathbf{x} - \overline{\mathbf{x}}|\tilde{\mathbf{y}}[i]] \tag{1.21}$$

where we denote $\tilde{\mathscr{A}}[t] = \sigma(\tilde{\mathbf{y}}[1], \cdots, \tilde{\mathbf{y}}[t])$ and $\overline{\mathbf{x}} := \mathbb{E}[\mathbf{x}]$.

Refer to [2] for the proof.

*Proof.* First, define the error:

$$\mathbf{e} := \mathbf{x} - \hat{E}[\mathbf{x}|\tilde{\mathscr{A}}[t]] = \mathbf{x} - \overline{\mathbf{x}} - \sum_{i=1}^{t} \hat{E}[\mathbf{x} - \overline{\mathbf{x}}|\tilde{\mathbf{y}}[i]]$$

We will show that $\hat{E}[\mathbf{x}|\tilde{\mathscr{A}}[t]]$ is the MMSE. By the Orthogonality Principle, this means we need to show that $\mathbf{e}$ is orthogonal to the linear span of $\{\tilde{\mathbf{y}}[1], \cdots, \tilde{\mathbf{y}}[t]\}$:

$$\mathbb{E}[\mathbf{e}^T(\tilde{\mathbf{y}}^T[1]c[1] + \tilde{\mathbf{y}}^T[2]c[2] + \cdots + \tilde{\mathbf{y}}^T[t]c[t] + b)] = 0$$

First, $\mathbb{E}[\mathbf{e}] = \mathbb{E}[\mathbf{x} - \overline{\mathbf{x}}] - \sum_{i=1}^{t} \mathbb{E}[\hat{E}[\mathbf{x} - \overline{\mathbf{x}}|\tilde{\mathbf{y}}[i]]]$, and $\mathbb{E}[\mathbf{x} - \overline{\mathbf{x}}] = 0$ and $\hat{E}[\mathbf{x} - \overline{\mathbf{x}}|\tilde{\mathbf{y}}[i]]$ is a linear function of $\tilde{\mathbf{y}}[i]$ for all $i = 1, \cdots, t$, which have zero mean by assumption.

Second, we must have $\mathrm{Cov}(\mathbf{e}, \tilde{\mathbf{y}}[j]) = 0$ for all $j = 1, 2, \cdots, t$. Starting from the left side:

$$\begin{aligned}
\mathrm{Cov}\left(\mathbf{x} - \overline{\mathbf{x}} - \sum_{i=1}^{k} \hat{E}[\mathbf{x} - \overline{\mathbf{x}}|\tilde{\mathbf{y}}[i]], \tilde{\mathbf{y}}[j]\right) &= \mathrm{Cov}(\mathbf{x} - \overline{\mathbf{x}}, \tilde{\mathbf{y}}[j]) - \sum_{i=1}^{k} \mathrm{Cov}(\hat{E}[\mathbf{x} - \overline{\mathbf{x}}|\tilde{\mathbf{y}}[i]], \tilde{\mathbf{y}}[j]) \\
&= \mathrm{Cov}(\mathbf{x}, \tilde{\mathbf{y}}[j]) - \mathrm{Cov}(\hat{E}[\mathbf{x} - \overline{\mathbf{x}}|\tilde{\mathbf{y}}[j]], \tilde{\mathbf{y}}[j]) \\
&= \mathrm{Cov}(\mathbf{x}, \tilde{\mathbf{y}}[j]) - \mathrm{Cov}(\mathrm{Cov}(\mathbf{x} - \overline{\mathbf{x}}, \tilde{\mathbf{y}}[j])\mathrm{Cov}(\tilde{\mathbf{y}}[j])^{-1}\tilde{\mathbf{y}}[j], \tilde{\mathbf{y}}[j]) \\
&= \mathrm{Cov}(\mathbf{x}, \tilde{\mathbf{y}}[j]) - \mathrm{Cov}(\mathbf{x}, \tilde{\mathbf{y}}[j])\mathrm{Cov}(\tilde{\mathbf{y}}[j])^{-1}\mathrm{Cov}(\tilde{\mathbf{y}}[j]) = 0
\end{aligned}$$

Together, the two imply that $\hat{E}[\mathbf{x}|\tilde{\mathscr{A}}[t]]$ is indeed the MMSE. ■

**Remark 3.** Although both the innovation and observation sequences span the same space ($\tilde{\mathscr{A}}[t] = \mathscr{A}[t]$), the elements of the innovations sequence are all orthogonal to each other. One might note the similarity of this construction process to the Gram-Schmidt orthonormalization process used in the context of linear algebra. □

**Remark 4.** We can simplify the expression (1.21) as follows:

$$\begin{aligned}
\hat{E}[\mathbf{x}|\tilde{\mathscr{A}}[t]] = \overline{\mathbf{x}} + \sum_{i=1}^{t} \hat{E}[\mathbf{x} - \overline{\mathbf{x}}|\tilde{\mathbf{y}}[i]] &= \overline{\mathbf{x}} + \sum_{i=1}^{t} \mathrm{Cov}(\mathbf{x}, \tilde{\mathbf{y}}[i])\mathrm{Cov}(\tilde{\mathbf{y}}[i])^{-1}\tilde{\mathbf{y}}[i] \\
&= \hat{E}[\mathbf{x}|\tilde{\mathscr{A}}[t-1]] + \hat{E}[\mathbf{x} - \overline{\mathbf{x}}|\tilde{\mathbf{y}}[t]]
\end{aligned} \tag{1.22}$$

This gives us a recursive formula in terms of each new observation $\tilde{\mathbf{y}}[t]$. □

### 1.4.2 The Discrete-Time Kalman Filter

For the sake of simplicity, we will initially consider the case where there is no control input.

$$\mathbf{x}[t+1] = A[t]\mathbf{x}[t] + \mathbf{w}[t] \tag{1.23a}$$
$$\mathbf{y}[t] = C[t]\mathbf{x}[t] + \mathbf{v}[t] \tag{1.23b}$$

where $\mathbf{x}[t], \mathbf{w}[t] \in \mathbb{R}^n, \mathbf{y}[t], \mathbf{v}[t] \in \mathbb{R}^m$, and $A[t] \in \mathbb{R}^{n\times n}, C[t] \in \mathbb{R}^{m\times n}$ are known for all $t \in \mathbb{N}$.

We make the following assumptions: $\mathbf{x}[0], \mathbf{v}[0], \mathbf{v}[1], \cdots, \mathbf{w}[0], \mathbf{w}[1], \cdots$ are all pairwise uncorrelated Gaussian-distributed random variables, and $\mathbb{E}[\mathbf{w}[t]] = 0, \mathrm{Cov}(\mathbf{w}[t]) = Q[t], \mathbb{E}[\mathbf{v}[t]] = 0, \mathrm{Cov}(\mathbf{v}[t]) = R[t]$ are known constant matrices. Further assume that $\mathbf{x}[0]$ comes from a known Gaussian distribution with $\mathbb{E}[\mathbf{x}[0]] = \bar{x}[0], \mathrm{Cov}(\mathbf{x}[0]) = P[0]$, and that it is pairwise uncorrelated with $\mathbf{w}[t]$ and $\mathbf{v}[t]$ for all $t$. Denote $\overline{\mathbf{x}}[t] = \mathbb{E}[\mathbf{x}[t]]$ and $P[t] = \mathrm{Cov}(\mathbf{x}[t])$. These quantities are recursively determined for $t \geq 1$ by:

$$\begin{aligned}\mathbb{E}[\mathbf{x}[t+1]] = A[t]\mathbb{E}[\mathbf{x}[t]] &\implies \overline{\mathbf{x}}[t+1] = A[t]\overline{\mathbf{x}}[t] \\ P[t+1] &= A[t]P[t]A[t]^T + Q[t]\end{aligned}$$

As before, we will define $\mathscr{A}[t] = \sigma(\mathbf{y}[0], \mathbf{y}[1], \cdots, \mathbf{y}[t])$ to represent the observations up until time $t$. Then we define $\hat{\mathbf{x}}[i|j] := \hat{E}[\mathbf{x}[i]|\mathscr{A}[j]]$ for nonnegative integers $i, j$, where $\hat{E}[\mathbf{x}[i]|\mathscr{A}_j]$ is the linear MMSE given by

$$\begin{aligned}\hat{E}[\mathbf{x}[i]|\mathbf{y}[j]] &= \mathbb{E}[\mathbf{x}[i]] + \mathrm{Cov}(\mathbf{x}[i], \mathbf{y}[j])\mathrm{Cov}(\mathbf{y}[j])^{-1}(\mathbf{y}[j] - \mathbb{E}[\mathbf{y}[j]]) \\ \implies \hat{E}[\mathbf{x}[i]|\mathscr{A}[j]] &= \hat{E}[\mathbf{x}[i]|\mathscr{A}[j-1]] + \mathrm{Cov}(\mathbf{x}[i], \mathbf{y}[j])\mathrm{Cov}(\mathbf{y}[j])^{-1}(\mathbf{y}[j] - \mathbb{E}[\mathbf{y}[j]]) \text{ by Remark 4}\end{aligned}$$

Finally, denote the associated covariance of error matrices $\hat{P}[i|j] := \mathrm{Cov}(\mathbf{x}[i] - \hat{\mathbf{x}}[i|j])$ for nonnegative integers $i, j$.

The goal is to compute an estimate of $\mathbf{x}[t]$ at each timestep $t$. We will do this by deriving a recursive relationship between successive state estimates $\hat{\mathbf{x}}[t-1|t-1]$ and $\hat{\mathbf{x}}[t|t]$.

The filtering process takes the same two steps as in the Bayesian framework from Section 1.1:

1. **Prediction**: we predict the value of $\hat{\mathbf{x}}[t|t-1]$ given $\hat{\mathbf{x}}[t-1|t-1]$. To do this, directly use the equations **(*)** and the fact that the noise random variables are uncorrelated from all the system variables. Equations yield:

$$\hat{\mathbf{x}}[t|t-1] = A[t-1]\hat{\mathbf{x}}[t-1|t-1] \tag{1.24a}$$
$$\hat{P}[t|t-1] = A[t-1]\hat{P}[t-1|t-1]A[t-1]^T + Q[t-1] \tag{1.24b}$$

2. **Measurement Update**: we modify our prediction from $\hat{\mathbf{x}}[t|t-1]$ to $\hat{\mathbf{x}}[t|t]$ in order to take into account a new observation $\mathbf{y}[t]$. Because we are able to predict a part of $\mathbf{y}[t]$ through the linear MMSE, the only innovation comes from the orthogonal component $\tilde{\mathbf{y}}[t] = \mathbf{y}[t] - \hat{E}[\mathbf{y}[t]|\mathscr{A}[t-1]]$. Alternatively written, this is:

$$\tilde{\mathbf{y}}[t] = \mathbf{y}[t] - C[t]\hat{\mathbf{x}}[t|t-1] \tag{1.25}$$

Derive $\hat{\mathbf{x}}[t|t]$ from $\hat{\mathbf{x}}[t|t-1]$:

$$\hat{\mathbf{x}}[t|t] = \hat{\mathbf{x}}[t|t-1] + \mathrm{Cov}(\mathbf{x}[t], \tilde{\mathbf{y}}[:t])\mathrm{Cov}(\tilde{\mathbf{y}}[t])^{-1}\tilde{\mathbf{y}}[t] \tag{1.26}$$

Furthermore, the covariance of error is updated:

$$\hat{P}[t|t] = \hat{P}[t|t-1] - \text{Cov}(\mathbf{x}[t], \tilde{\mathbf{y}}[t])\text{Cov}(\tilde{\mathbf{y}}[t])^{-1}\text{Cov}(\tilde{\mathbf{y}}[t], \mathbf{x}[t]) \tag{1.27}$$

Intuitively, the use of the new observation $\tilde{\mathbf{y}}[t]$ reduces the covariance of error for predicting $\mathbf{x}[t]$ from $\hat{P}[t|t-1]$ by the covariance matrix of the innovative part of the estimator.

Let us define the gain $L[t] := \text{Cov}(\mathbf{x}[t], \tilde{\mathbf{y}}[:t])\text{Cov}(\tilde{\mathbf{y}}[t])^{-1}$ so that we can simplify the information update equations as:

$$\begin{aligned} \hat{\mathbf{x}}[t|t] &= \hat{\mathbf{x}}[t|t-1] + L[t]\tilde{\mathbf{y}}[t] && (1.28a) \\ \hat{P}[t|t] &= \hat{P}[t|t-1] - L[t]\text{Cov}(\tilde{\mathbf{y}}[t], \mathbf{x}[t]) && (1.28b) \end{aligned}$$

We can further calculate

$$\begin{aligned} \text{Cov}(\mathbf{x}[t], \tilde{\mathbf{y}}[t]) &= \hat{P}[t|t-1]C[t]^T \\ \text{Cov}(\tilde{\mathbf{y}}[t]) &= C[t]\hat{P}[t|t-1]C[t]^T + R[t] \end{aligned}$$

so that:

$$L[t] = \hat{P}[t|t-1]C[t]^T(C[t]\hat{P}[t|t-1]C[t]^T + R[t])^{-1} \tag{1.29}$$

Combining the results of the information update together with the time update, we obtain our final equations:

$$\begin{aligned} \hat{\mathbf{x}}[t|t] &= A[t-1]\hat{\mathbf{x}}[t-1|t-1] + L[t]\tilde{\mathbf{y}}[t] && (1.30a) \\ \hat{P}[t|t] &= \hat{P}[t|t-1] - L[t]C[t]\hat{P}[t|t-1] = (I - L[t]C[t])(A[t-1]\hat{P}[t-1|t-1]A[t-1]^T + Q[t-1]) && (1.30b) \end{aligned}$$

The alternative, more common form of the Kalman filter equations is the update process from $\mathbf{x}[t|t-1]$ to $\mathbf{x}[t+1|t]$:

$$\begin{aligned} \hat{\mathbf{x}}[t+1|t] &= A[t](\hat{\mathbf{x}}[t|t-1] + L[t]\tilde{\mathbf{y}}[t]) \\ \hat{P}[t+1|t] &= A[t](\hat{P}[t|t-1] - L[t]\text{Cov}(\tilde{\mathbf{y}}[:t], \mathbf{x}[t]))A[t]^T + Q[t] \end{aligned}$$

where $L[t]$ follows as in (1.32). In a later section, we will discuss the specialization to Poisson-distributed disturbances, which also shares some of the key properties that make the analysis easy. These noise characteristics, which will be detailed formally later, have to do with the fact that both Gaussian and Poisson types of noise are Lévy.

**Theorem 4** (The Discrete-Time Kalman Filtering Equations)**.** The DTKF equations are given as follows:

$$\begin{aligned} \hat{\mathbf{x}}_{k|k} &= A_{k-1}\hat{\mathbf{x}}_{k-1|k-1} + L_k\tilde{\mathbf{y}}_k && (1.31a) \\ \Sigma_{k|k} &= \Sigma_{k|k-1} - L_kC_k\Sigma_{k|k-1} = (I - L_kC_k)(A_{k-1}\Sigma_{k-1|k-1}A_{k-1}^T + Q_{k-1}) && (1.31b) \end{aligned}$$

where

$$L_k = \Sigma_{k|k-1}C_k^T(C_k\Sigma_{k|k-1}C_k^T + R_k)^{-1} \tag{1.32}$$

The proof to this result can be found in any standard control-theoretic textbook. In particular, one which uses the linear innovation sequence derived in the previous section is given in [2].

The alternative, more common form of the Kalman filter equations is the update process from $\mathbf{x}_{k|k-1}$ to $\mathbf{x}_{k+1|k}$:

$$\begin{aligned}\hat{\mathbf{x}}_{k+1|k} &= A_k(\hat{\mathbf{x}}_{k|k-1} + L_k\tilde{\mathbf{y}}_k)\\ \Sigma_{k+1|k} &= A_k(\Sigma_{k|k-1} - L_k\mathrm{Cov}(\tilde{\mathbf{y}}^k, \mathbf{x}_k))A_k^T + Q_k\end{aligned}$$

### 1.4.3 The Continuous-Time Kalman Filter

Now consider the continuous-time dynamics

$$\begin{aligned}\dot{\mathbf{x}}(t) &= A\mathbf{x}(t) + B_w\mathbf{w}(t) && \text{(CT)}\\ \mathbf{y}(t) &= C\mathbf{x}(t) + \mathbf{v}(t) && (1.33)\end{aligned}$$

where $\mathbf{x}(t), \mathbf{w}(t) \in \mathbb{R}^n, \mathbf{y}(t), \mathbf{v}(t) \in \mathbb{R}^m$, and $A, B_w \in \mathbb{R}^{n\times n}, C \in \mathbb{R}^{m\times n}$ are known for all $t > 0$.

As in the discrete-time case, we make the following assumptions: $\mathbf{x}(0) := \mathbf{x}_0$, $\{\mathbf{w}(t)\}$, and $\{\mathbf{v}(t)\}$ are pairwise uncorrelated for all $t > 0$, and $\mathbb{E}[\mathbf{w}(t)] = 0, \mathrm{Cov}(\mathbf{w}(s), \mathbf{w}(t)) = \mathbb{E}[\mathbf{w}(s)\mathbf{w}(t)] = Q\delta(t-s), \mathbb{E}[\mathbf{v}(t)] = 0, \mathrm{Cov}(\mathbf{v}(s), \mathbf{v}(t)) = R\delta(t-s)$, where $Q$ and $R$ are known constant matrices. Further assume that $\mathbf{x}_0$ comes from a known Gaussian distribution with $\mathbb{E}[\mathbf{x}_0] = 0$ (assumed for simplicity), $\mathrm{Cov}(\mathbf{x}_0) = \Sigma_0$. We will define $\mathscr{A}(t) = \sigma\{\mathbf{y}(s) \,:\, 0 \le s < t\}$ to represent the observations made until time $t$.

The goal is to estimate $\hat{\mathbf{x}}(t)$ of $\mathbf{x}(t)$ given the observations $\mathscr{A}(t)$ such that the MSE:

$$J := \mathbb{E}\left[\mathrm{tr}((\mathbf{x}(t) - \hat{\mathbf{x}}(t))(\mathbf{x}(t) - \hat{\mathbf{x}}(t))^T)\right] \tag{1.34}$$

is minimized. In analogue to the DTKF, the MMSE $\hat{\mathbf{x}}(t)$ is given by $\mathbb{E}[\mathbf{x}(t) \mid \mathscr{A}(t)]$. This implies that $\hat{\mathbf{x}}(0) = \mathbb{E}[\mathbf{x}_0] = 0$. Additionally, one can derive the dynamics:

$$\dot{\hat{\mathbf{x}}}(t) = A\hat{\mathbf{x}}(t) + K(\mathbf{y} - C\hat{\mathbf{x}}(t))$$

For the moment, we will take this as given, and use it to derive the covariance equation and the optimal Kalman filter gain.

Define the error vector $\mathbf{e} = \mathbf{x} - \hat{\mathbf{x}}$. Its dynamics are given by:

$$\dot{\mathbf{e}} = \dot{\mathbf{x}} - \dot{\hat{\mathbf{x}}} = A\mathbf{x} + B_w\mathbf{w} - A\hat{\mathbf{x}} - K(\mathbf{y} - C\hat{\mathbf{x}}) = (A - KC)\mathbf{e} + B_w\mathbf{w} - K\underbrace{(\mathbf{y} - C\mathbf{x})}_{\mathbf{v}}$$

Define the error covariance $\Sigma := \mathbb{E}[\mathbf{e}\mathbf{e}^T]$. Then note that the MSE at time $t$ is exactly $\mathrm{tr}(P)$. Derive the dynamics of $\Sigma$ as follows:

$$\begin{aligned}\dot{\Sigma} &= \mathbb{E}[\dot{\mathbf{e}}\mathbf{e}^T + \mathbf{e}\dot{\mathbf{e}}^T]\\ &= \mathbb{E}[(A - KC)\mathbf{e}\mathbf{e}^T + \mathbf{e}\mathbf{e}^T(A^T - C^TK^T)] + \mathbb{E}[B_w\mathbf{w}\mathbf{e}^T + \mathbf{e}\mathbf{w}^TB_w^T] + \mathbb{E}[-K\mathbf{v}\mathbf{e}^T - \mathbf{e}\mathbf{v}^TK]\\ &= (A - KC)\Sigma + \Sigma(A^T - C^TK^T) + \mathbb{E}[B_w\mathbf{w}\mathbf{e}^T + \mathbf{e}\mathbf{w}^TB_w^T] + \mathbb{E}[-K\mathbf{v}\mathbf{e}^T - \mathbf{e}\mathbf{v}^TK^T] \qquad (1.35)\end{aligned}$$

Denote the state transition matrix $\Phi(t_0, t) := e^{(A-KC)(t-t_0)}$. Then we can write $\mathbf{e}(t)$ as:

$$\mathbf{e}(t) = \Phi(0,t)\mathbf{e}_0 + \int_0^t \Phi(s,t)B_w\mathbf{w}(s)ds - \int_0^t \Phi(s,t)K\mathbf{v}(s)ds \tag{1.36}$$

which implies that

$$\begin{aligned}
\mathbb{E}[\mathbf{e}\mathbf{w}^T(t)B_w^T] &= \Phi(0,t)\mathbb{E}[\mathbf{e}_0\mathbf{w}^T]B_w^T + \int_0^t \Phi(s,t)B_w\mathbb{E}[\mathbf{w}(s)\mathbf{w}^T(t)]B_w^T ds - \int_0^t \Phi(s,t)K\mathbb{E}[\mathbf{w}(s)\mathbf{v}^T(t)]B_w^T ds \\
&= \int_0^t \Phi(s,t)B_wQ\delta(t-s)B_w^T ds \\
&= \frac{1}{2}B_wQB_w^T \text{ since } \Phi(t,t) = I
\end{aligned}$$

One can make a symmetric argument for $\mathbb{E}[B_w\mathbf{w}\mathbf{e}^T] = \frac{1}{2}B_wQB_w^T$.

For the last line, we also have the following formula:

$$\int_a^b f(x)\delta(b-x)dx = \int_{b-a}^0 f(b-u)\delta(u)du = \frac{1}{2}f(b)$$

Informally speaking, since the delta function occurs at one of the endpoints of the integral, only half the total weight gets integrated over.

Similarly, (1.36) implies that $\mathbb{E}[-\mathbf{e}\mathbf{v}^T(t)K^T] = \mathbb{E}[-K\mathbf{v}\mathbf{e}^T] = (1/2)KRK^T$ since the minus signs cancel.

Substituting everything back int (1.35) yields:

$$\begin{aligned}
\dot{\Sigma} &= (A-KC)\Sigma + \Sigma(A^T - C^TK^T) + B_wQB_w^T + KRK^T \\
&= A\Sigma + \Sigma A^T + B_wQB_w^T - KC\Sigma - \Sigma C^TK^T + KRK^T \quad (1.37)\\
&= A\Sigma + \Sigma A^T + B_wQB_w^T + (KR - \Sigma C^T)R^{-1}(KR - \Sigma C^T)^T - \Sigma C^TRC\Sigma \quad (1.38)
\end{aligned}$$

where the last line follows from completing the square (exercise). To minimize $J = \text{tr}(\Sigma(t))$, we can minimize $\Sigma(t)$ by choosing $K$ so that $\dot{P}(t)$ decreases by the maximum amount possible at time $t$. This happens when the term $(KR - \Sigma C^T)R^{-1}(KR + \Sigma C^T)^T$ is equal to 0, since it can never be negative (like a square term).

This implies:

$$K(t)R - \Sigma(t)C^T = 0 \Longrightarrow K(t) = \Sigma(t)C^TR^{-1}$$

Substituting this back into (1.37) yields the final covariance equation.

# Chapter 2

# Multi-Sensor Bayesian Filtering

The general Bayesian filtering framework from Section 1.1 was designed for a single sensor tracking the state of a single target. For multiple sensors and multiple targets, a similar but recursive framework is used. Individual targets are allocated across the state space $\mathcal{X} := \mathbb{R}^{n_x}$. In addition to the usual movement of targets in the space via dynamics similar to those of the single-target case (1.5), we have at each timestep the 1) possible disappearance of existing targets, and 2) the spawning of new targets from existing parent targets, in a manner similar to branching processes. Thus, the total number of targets at time $k$, $N_k \in \mathbb{N}$, is a time-varying quantity.

To represent the multi-target state space, we denote $\mathcal{X}^{\cup} := \cup_{m\in\mathbb{N}}\mathcal{X}^m$ to be the union of all $\mathcal{X}$-valued sequences of all lengths $m \in \mathbb{N}$. The new overall state that we want to estimate is denoted in upper-case $X_k := \{\mathbf{x}_{1,k}, \mathbf{x}_{2,k}, \cdots, \mathbf{x}_{N_k,k}, \cdots\} \in \mathcal{X}^{\cup}$ for each time $k$.

The dynamics for the multi-sensor, multi-target case vary according to stochastic dynamics. As in Section 1.1, we assume a Markovian setting. First, we again denote $p(\mathbf{x}_{k+1}|\mathbf{x}_k)$ to be the single-target probability of transitioning from state $\mathbf{x}_k$ to state $\mathbf{x}_{k+1}$ for any $\mathbf{x}_k, \mathbf{x}_{k+1} \in \mathcal{X}$. To keep track of the births of new targets and the deaths of existing targets, we use the following st notations

$$X_{k+1} = \Theta(X_k)\Psi(X_k) := \{\theta(\mathbf{x}_{1,k}) \cup \cdots \theta(\mathbf{x}_{N_k,k})\} \cup \{\psi(\mathbf{x}_{1,k}) \cup \cdots \psi(\mathbf{x}_{N_k,k})\} \tag{2.1}$$

where $\theta(\mathbf{x}_{i,k})$ represents whether or not the target with state $\mathbf{x}_{i,k}$ has survived until the next timestep

$$\theta(\mathbf{x}_{i,k}) = \begin{cases} \{\mathbf{x}_{i,k}\} & \text{w.p. } q_S \\ \varnothing & \text{w.p. } 1 - q_S \end{cases}$$

where $q_S$ denotes the probability of survival. $\psi(\mathbf{x}_{i,k})$ is the set of targets that were spawned from $\mathbf{x}_{i,k}$

$$\psi(\mathbf{x}_{i,k}) = X_{j,k+1} \text{ w.p. } b(X_{j,k}|\mathbf{x}_{i,k})$$

for $X_{j,k+1} \subset X_{k+1}$ and $b(X_{j,k}|\mathbf{x}_{i,k})$ denoting the probability of spawning new target set $X_{j,k}$ from state $\mathbf{x}_{i,k}$.

Let $M$ be the number of sensors. We denote $\mathscr{B}_k^{(j)} := \sigma(Y_1^{(j)}, \cdots, Y_k^{(j)})$, where $Y_k := \{Y_k^{(1)}, Y_k^{(2)}, \cdots, Y_k^{(M)}\}$ denotes the observation set of all sensors at time $k$, and $Y_k^{(j)} := \{\mathbf{y}_{1,k}^{(j)}, \mathbf{y}_{2,k}^{(j)}, \cdots\}$ denotes the observation set of sensor $j$ at time $k$. We have each individual vector measurement $\mathbf{y}_{m,k}^{(j)}$ belonging in $\mathcal{Y}$, and we define $\mathcal{Y}^{\cup}$ in the same way as $\mathcal{X}^{\cup}$. To clarify the notation further, we remark that $Y_k$ is not equivalent to

$\mathscr{A}_k := \sigma(\mathbf{y}_1, \cdots, \mathbf{y}_k)$, which describes the single-sensor sequence of observations up to time $k$ instead of multiple sensors' observations at a single time $k$.

The measurement equation for the multi-target, multi-sensor case is also different from the single-target, single-sensor case. In addition to additive noise and vector projection from $\mathcal{X}$ to $\mathcal{Y}$, stochasticity in measurements also comes from the possibility of picking up false alarms, i.e. clutter targets. Due to the spatial allocation of sensors, some targets may not be observed at all due to the distance.

$$Y_k^{(j)} = \Xi^{(j)}(X_k) \cup \Gamma_k^{(j)} = \xi^{(j)}(\mathbf{x}_{1,k}) \cup \cdots \cup \xi^{(j)}(\mathbf{x}_{N_k,k}) \cup \Gamma_k^{(j)} \tag{2.2}$$

where $\Gamma_k^{(j)}$ denotes the set of clutter targets that are picked up by sensor $j$ at time $k$. $\xi^{(j)}(\mathbf{x}_k)$ denotes the observation set of sensor $j$ produced by the target $\mathbf{x}_k$

$$\xi^{(j)}(\mathbf{x}_k) = \begin{cases} \{\mathbf{y}_k^{(j)}\} & \text{w.p. } q_D^{(j)}(\mathbf{x}_k)\ell^{(j)}(\mathbf{y}_k^{(j)}|\mathbf{x}_k) \\ \varnothing & \text{w.p. } 1 - q_D^{(j)}(\mathbf{x}_k) \end{cases}$$

where $q_D^{(j)}(\mathbf{x}_k)$ denotes the probability of sensor $j$ being able to detect target state $\mathbf{x}_k$, and $\ell^{(j)}(\mathbf{y}_k^{(j)}|\mathbf{x}_k)$ denotes the likelihood probability of sensor $j$ detecting measurement $\mathbf{y}_k^{(j)}$ from $\mathbf{x}_k$.

We solve the filtering problem according to the alternating two-step approach of prediction and measurement update. Note that the integral expressions in (1.6) and (1.7) no longer work for the multitarget scenario because the state $X_k$ is a set. We define the set integral as follows.

**Definition 3** (Set Integral)**.** The set integral of a function $g : \mathcal{S}^\cup \to \mathbb{R}$, where $\mathcal{S}^\cup := \cup_{n=1}^{\infty}\mathcal{S}$ for any set $\mathcal{S}$ is given by

$$\int_{\mathcal{S}^\cup} g(X)\delta X := g(\varnothing) + \sum_{n=0}^{\infty} \frac{1}{n!} \int_{\mathcal{S}^n} g(\{\mathbf{x}_1, \cdots, \mathbf{x}_n\}) d\mathbf{x}_1 \cdots d\mathbf{x}_n$$

□

The two steps are then given as follows:

1. **Prediction**: given $f(X_k|\mathscr{B}_k)$, predict $f(X_k|\mathscr{B}_k)$ using Chapman-Kolmogorov equation

$$f(X_{k+1}|\mathscr{B}_k) = \int_{\mathcal{X}^\cup} f(X_{k+1}|X_k, \mathscr{B}_k) f(X_k|\mathscr{B}_k)\delta X_k = \int_{\mathcal{X}^\cup} p(X_{k+1}|X_k) f(X_k|\mathscr{B}_k)\delta X_k \tag{2.3}$$

   where $p(X_{k+1}|X_k)$ is the transition probability from state-set $X_k$ to state-set $X_{k+1}$, defined similarly to the single-target version $p(\mathbf{x}_{k+1}|\mathbf{x}_k)$.

2. **Measurement Update**: given $f(X_{k+1}|\mathscr{B}_k)$, incorporate new measurement $Y_{k+1}$ to get $f(X_{k+1}|\mathscr{B}_{k+1})$

$$f(X_{k+1}|\mathscr{B}_{k+1}) = \frac{f(X_{k+1}, Y_{k+1}|\mathscr{B}_k)}{f(Y_{k+1}|\mathscr{B}_k)} = \frac{\ell(Y_{k+1}|X_{k+1}) f(X_{k+1}|\mathscr{B}_k)}{f(Y_{k+1}|\mathscr{B}_k)} \tag{2.4}$$

   where $\ell(Y|X)$ denotes the likelihood of observing observation set $Y$ from state-set $X$, defined similarly to the single-target version $\ell(\mathbf{y}|\mathbf{x})$.

We consider the multitarget, multisensor analog to the Kalman filter: namely, we choose a specific first-order moment statistic and propagate that instead of the entire posterior distribution. This type of filter was introduced by [3].

## 2.1 Change of Measure for the Poisson Process

The expression for the Doléans-Dade exponential (1.13) might be familiar to the reader, especially when viewing it as an analogy to the well-known fact that the form of the solution to the equation $dZ(t) = Z(t)dt, Z(0) = 1$ is given by the exponential $Z(t) = e^t$. More generally, for ny scalar-valued, differentiable, deterministic function $X(t)$, the solution to the system $dZ(t) = Z(t)dX(t), Z(0) = 1$ is given by $Z(t) = e^{(X(t)-X(0))}$. Clearly, when $dX(t) = \mu(t)dW(t)$, we obtain (1.13) as the solution to $dZ(t) = Z(t)\mu(t)dW(t), Z(0) = 1$, which was shown in Proposition 1 when $\mu(t) = \mu$, a constant. Likewise, when $dX(t) = \mu^T(t)dW(t)$, we obtain (1.14), as the solution to $dZ(t) = Z(t)\mu^T(t)dW(t), Z(0) = 1$. Recall that the interpretation of $Z(t)$ is that it is the Radon-Nikodym derivative used to transform from $dW(t)$ to $d\tilde{W}(t) := dW(t) - \mu(t)dt$.

**Proposition 2.** The Doléans-Dade exponential for semimartingale $X$ is given by:

$$Z(t) = e^{X(t)-\frac{1}{2}[X,X](t)} \prod_{0<s\leq t} (1 + \Delta X(s))e^{-\Delta X(s)+\frac{1}{2}\Delta X^2(s)} \tag{2.5}$$

Note that the jump part of the quadratic variation cancels out with the jumps parts in the product term. This allows us to rewrite the expression as:

$$Z(t) = e^{X(t)-\frac{1}{2}[X,X]^c(t)} \prod_{0<s\leq t} (1 + \Delta X(s))e^{-\Delta X(s)}$$

*Proof of Proposition 2.* The system $\Delta Z_n = Z_{n-1}\Delta X_n, Z_0 = 1$, with $\Delta Z_n := Z_n - Z_{n-1}$ can be recursively substituted backwards until the base case. We get:

$$Z_n = (1 + \Delta X_n)Z_{n-1} = \prod_{i=1}^{n}(1 + \Delta X_i)$$

This is easily extendable to the continuous-time case:

$$Z(t) = (1 + \Delta X(t))Z(t-) \text{ for all } t > 0$$

Apply Itô's formula for semimartingales:

$$d\left(\ln(Z(t))\right) = \frac{dZ(t)}{Z(t)} - \frac{1}{2Z^2(t)}d[Z,Z]^c(t) + \sum_{0<s\leq t}\left[\ln(Z(s)) - \ln(Z(s-)) - \frac{\Delta Z(s)}{Z(s-)}\right] \tag{2.6}$$

where $\Delta Z(s) := Z(s) - Z(s-)$.

We have $d[Z,Z]^c(t) = Z^2 d[X,X]^c(t)$ and

$$\ln(Z(s)) - \ln(Z(s-)) = \ln\left(\frac{Z(s-)(1 + \Delta X(s))}{Z(s-)}\right) = \ln(1 + \Delta X(s))$$
$$\frac{\Delta Z(s)}{Z(s-)} = \frac{Z(s-)\Delta X(s)}{Z(s-)} = \Delta X(s)$$

Substituting, we get:

$$(2.6) = dX(t) - \frac{1}{2}d[X,X]^c(t) + \sum_{0<s\leq t}\left[\ln(1 + \Delta X(s)) - \Delta X(s)\right]$$

$$\Longrightarrow Z(t) = e^{X(t)-\frac{1}{2}[X,X]^c(t)} \prod_{0<s\leq t} (1+\Delta X(s))e^{-\Delta X(s)}$$

which is exactly our desired formula. ■

How does the Doléans-Dade exponential look like for specific Poisson processes? In the following two propositions, we show the form of the Radon-Nikodym derivatives needed for the standard Poisson process and the compound Poisson process.

**Proposition 3** (CoM for the Standard Poisson Process)**.** Suppose $N(t)$ is a standard Poisson process with constant intensity $\lambda$ according to measure $P$. Then the Radon-Nikodym derivative to transform $N(t)$ to a standard Poisson process with intensity $\tilde{\lambda}$ is expressed as

$$Z(t) = e^{-t(\lambda-\tilde{\lambda})} \left(\frac{\tilde{\lambda}}{\lambda}\right)^{N(t)} \tag{2.7}$$

Furthermore, (2.7) satisfies the SDE

$$dZ(t) = Z(t-) \left(\frac{\tilde{\lambda}-\lambda}{\lambda}\right) d(N(t)-\lambda t) \tag{2.8}$$

*Proof.* First, it is easy to verify (2.7) by writing out the probability distributions:

$$Q(N(t)=k) = e^{-t(\tilde{\lambda}-\lambda)} \left(\frac{\tilde{\lambda}}{\lambda}\right)^k P(N(t)=k) = e^{-t(\tilde{\lambda}-\lambda)} \left(\frac{\tilde{\lambda}}{\lambda}\right)^k \cdot e^{-\lambda t}\frac{(\lambda t)^k}{k!} = e^{-\tilde{\lambda}}\frac{\tilde{\lambda}^k t^k}{k!}$$

Thus, $N(t)$ is also Poisson under the measure $Q$, just with intensity $\tilde{\lambda}$ as opposed to $\lambda$.

Note that

$$X^c(t) = (\tilde{\lambda}-\lambda)t,\ X^d(t) = \left(\frac{\tilde{\lambda}-\lambda}{\lambda}\right) N(t),\ [X,X]^c(t) = 0$$

Hence, if there exists a jump at time $t$, $\Delta X(t) = \Delta X^d(t)$. Rearranging the terms yields

$$\frac{\tilde{\lambda}}{\lambda} = 1 + \Delta X(t)$$

Applying these to Proposition 2, we get

$$Z(t) = e^{\left(\frac{\tilde{\lambda}}{\lambda}-1\right)(N(t)-\lambda t)-0} \prod_{0<s\leq t} \left(\frac{\tilde{\lambda}}{\lambda}\right) e^{-\left(\frac{\tilde{\lambda}}{\lambda}\right)} = e^{(\lambda-\tilde{\lambda})t} \left(\frac{\tilde{\lambda}}{\lambda}\right)^{N(t)}$$

and we indeed obtain the desired expression (2.7). ■

Now consider the compound Poisson process $Y(t)$, written as follows:

$$Y(t) = \sum_{i=1}^{N(t)} \xi_i \tag{2.9}$$

where $N(t)$ is the standard Poisson process of intensity $\lambda$, and the jump sizes $\xi_i$ are iid random variables.

It is of convenience to recall the moment-generating function for (2.9).

**Proposition 4** (Moment-Generating Function of Compound Poisson Process)**.** The moment-generating function of $Y(t)$ from (2.9) is given by

$$\phi_Y(t) = e^{\lambda t \int_{-\infty}^{\infty} (e^{u\xi}-1)\nu(d\xi)}$$

*Proof.* This can be seen through the following argument. Since $N(t)$ has a Poisson distribution with parameter $t > 0$ and is independent of $\{\xi_k\}$, we get

$$\mathbb{E}_P\left[e^{uY(t)}\right] = \mathbb{E}_P\left[e^{\left(u\sum_{k=1}^{N(t)}\xi_k\right)}\right] = \sum_{n=0}^{\infty}\mathbb{E}_P\left[e^{\left(u\sum_{k=1}^{n}\xi_k\right)}\middle| N(t)=n\right]P(N(t)=n) \tag{2.10}$$

where the last equality follows by definition of expectation. From the fact that the Poisson process has stationary increments:

$$\begin{aligned}(2.10) &= e^{-\lambda t}\sum_{n=0}^{\infty}\frac{(\lambda t)^n}{n!}\mathbb{E}_P\left[e^{\left(u\sum_{k=1}^{N(T)-N(t)}\xi_k\right)}\right]\\ &= e^{-\lambda t}\sum_{n=0}^{\infty}\frac{(\lambda t)^n}{n!}\prod_{k=1}^{n}\mathbb{E}_P\left[e^{u\xi_1}\right]^n \quad \text{since } \xi_k \text{ iid}\\ &= e^{-\lambda t}e^{\lambda t\mathbb{E}_P[e^{u\xi_k}]} \quad \text{by Taylor expansion}\\ &= e^{\lambda t\left(\mathbb{E}_P[e^{u\xi_1}]-1\right)} = e^{\lambda t\int_{-\infty}^{\infty}(e^{uy}-1)\nu(dy)}\end{aligned}$$

where the last equality comes from the fact that $\int_{-\infty}^{\infty}\nu(dy) = 1$. ∎

Now we investigate the change of measure formula for the compound Poisson process. We show below that it simply becomes a multiplication of standard Poisson processes grouped by jumps of the same height. The interpretation is that we are shifting each type of jump individually.

**Proposition 5** (CoM for the Compound Poisson Process)**.** Suppose we are given a compound Poisson process of the form (2.9), with constant intensity $\lambda$ and jumps $\xi_1, \xi_2, \cdots$ with distribution $f(\xi)$ (measure $\nu(d\xi) := f(\xi)d\xi$) under measure $P$. Then the Radon-Nikodym derivative used to transform (2.9) to a compound Poisson process with intensity $\tilde{\lambda}$ and jump distribution $\tilde{f}(\xi)$ (measure $\tilde{\nu}(d\xi) := \tilde{f}(\xi)d\xi$) is given by

$$Z(t) = e^{(\lambda-\tilde{\lambda})t}\prod_{i=1}^{N(t)}\frac{\tilde{\lambda}\tilde{f}(\xi_i)}{\lambda f(\xi_i)} \tag{2.11}$$

Moreover, the process (2.11) satisfies the SDE

$$dZ(t) = Z(t-)\left(d\left(\sum_{i=1}^{N(t)}\frac{\tilde{\lambda}\tilde{f}(\xi_i)}{\lambda f(\xi_i)} - \tilde{\lambda}t\right) - d(N(t)-\lambda t)\right) \tag{2.12}$$

*Proof.* Define the two quantities

$$J(t) := \prod_{i=1}^{N(t)}\frac{\tilde{\lambda}\tilde{f}(\xi_i)}{\lambda f(\xi_i)}, \quad H(t) := \sum_{i=1}^{N(t)}\frac{\tilde{\lambda}\tilde{f}(\xi_i)}{\lambda f(\xi_i)}$$

Then we can write the recursive relationships

$$J(t) = J(t-) \cdot \frac{\tilde{\lambda}\tilde{f}(\xi_{N(t)})}{\lambda f(\xi_{N(t)})} = J(t-) \cdot \frac{\tilde{\lambda}\tilde{f}(\Delta Y(t))}{\lambda f(\Delta Y(t))}$$

$$\Longrightarrow \quad \Delta J(t) = J(t-) \left( \frac{\tilde{\lambda}\tilde{f}(\Delta Y(t))}{\lambda f(\Delta Y(t))} - 1 \right)$$

and

$$H(t) = H(t-) + \frac{\tilde{\lambda}\tilde{f}(\Delta Y(t))}{\lambda f(\Delta Y(t))} \implies \Delta H(t) = \frac{\tilde{\lambda}\tilde{f}(\Delta Y(t))}{\lambda f(\Delta Y(t))}$$

We then obtain a SDE for $J(t)$ as follows:

$$\Delta J(t) = J(t-)(\Delta H(t) - 1) = J(t-)(\Delta H(t) - \Delta N(t)) \; dJ(t) = J(t-)(dH(t) - dN(t))$$

We can now prove (2.12) using Itô's formula to get $dZ(t)$, with $Z(t) = e^{(\lambda - \tilde{\lambda})t} J(t)$.

$$\begin{aligned} Z(t) &= Z(0) + \int_0^t (\lambda - \tilde{\lambda}) e^{(\lambda-\tilde{\lambda})s} J(s-) ds + \int_0^t e^{(\lambda-\tilde{\lambda})s} dJ(s) \\ &= Z(0) + (\lambda - \tilde{\lambda}) \int_0^t \underbrace{e^{(\lambda-\tilde{\lambda})s} J(s-)}_{=:Z(s-)} ds + \int_0^t \underbrace{e^{(\lambda-\tilde{\lambda})s} J(s-)}_{=:Z(s-)} dH(s) - \int_0^t \underbrace{e^{(\lambda-\tilde{\lambda})s} J(s-)}_{=:Z(s-)} dN(s) \end{aligned}$$

and hence, we confirm (2.12):

$$dZ(t) = Z(t-) d\left( (H(t) - \tilde{\lambda}t) - (N(t) - \lambda t) \right)$$

To show that the transformation indeed yields another Poisson process, we use characteristic functions. Define

$$G(t) := e^{uY(t) - \tilde{\lambda}t(\tilde{\phi}_\xi(u) - 1)}$$

where $\tilde{\phi}_\xi(u) := \mathbb{E}_Q[e^{u\xi}] = \int_{-\infty}^{\infty} e^{u\xi} \tilde{\nu}(d\xi)$ is the characteristic function of the jumps $\xi$ under measure $Q$. It is possible to show that $G(t)Z(t)$ is a martingale. This implies

$$\mathbb{E}_P[G(t)Z(t)] = 1 \implies e^{-\tilde{\lambda}t(\tilde{\phi}_\xi(u)-1)} \mathbb{E}_P\left[e^{uY(t)} Z(t)\right] = 1 \implies \mathbb{E}_Q\left[e^{uY(t)}\right] = \mathbb{E}_P\left[e^{uY(t)} Z(t)\right] = e^{\tilde{\lambda}t(\tilde{\phi}_\xi(u)-1)}$$

which shows that the distribution of $Y(t)$ is indeed a compound Poisson with intensity $\tilde{\lambda}$ and jumps $\tilde{f}(\xi)$ under measure $Q$. ■

## 2.2 Hierarchical Architectures for Sensor Fusion

The problem of *multisensor data fusion* is understood as a reference to techniques which combine data about the state of an environment or a system from multiple external sensors for the purposes of drawing better inferences, or deriving a more accurate estimate about the state. In the multisensor fusion literature [4, 5], there are two common *hierarchical architectures* depending on the way computational resources are allocated across nodes: 1) a *centralized fusion* scheme, and 2) a *distributed fusion* scheme. There is also a *decentralized fusion* scheme, in which each node performs fusion indpendently using information retrieved from its neighbors. In contrast to hierarchical architectures, there is no single point of fusion in decentralized architectures and nodes are often treated as equals. We will not consider decentralized fusion; in this section, out focus is on hierarchical architectures such as the centralized and distributed fusion approaches.

In hierarchical data fusion, we distinguish between *lower-level nodes* and *higher-level nodes*. Lower-level nodes are equipped with external sensors which can obtain state information directly from the system or environment, while higher-level nodes do not nodes have external sensors and only communicate with the lower-level nodes.

### 2.2.1 Two-Node Motivation: Bayesian Framework

Consider the data fusion problem with two lower-level nodes and a higher-level node.

- in the centralized fusion scheme, the lower-level nodes forward exactly the unprocessed sensor observations to the higher-level node, which constructs the overall estimate of the state based on the observations. Very little computational resources are required to power the lower-level nodes since they are simply forwarding what their sensors collect; much of the burden falls on the higer-level node.
- in the distributed fusion scheme, the lower-level nodes perform some processing with the sensor observations, and they each construct their own local estimate of the state before transmitting it to the higher-level node. Compared to centralized fusion, more computational resources should be devoted to the lower-level nodes; in contrast, the higher-level node requires less power since it only needs to fuse the two local estimates to construct the overall estimate of the state.

**Assumption 1.** For the moment, we do not consider communication delays, bandwidth constraints, power constraints, finite memory, and other typical limiting factors which come into play for algorithms which are deployed as a network consisting of multiple components. □

We can extend the Bayesian formulation of Section 1.1 to the hierarchical data fusion case with three nodes. We refer to the two lower-level nodes as Nodes 1 and 2 and the higher-level node as Node 0. Denote $\mathbf{x} \in \mathbb{R}^n$ to be the vector representing the true state, and let $\mathcal{Y}_j = \{\mathbf{y}_j[1], \mathbf{y}_j[2], \mathbf{y}_j[3], \cdots\} \subset \mathbb{R}^{m_j}$ be the set of measurements corresponding to Node $j$, $j \in \{1, 2\}$.

**Assumption 2.** We have conditional independence over all the measurements given the true state:

$$p(\mathbf{y}_j[s], \mathbf{y}_k[t]|\mathbf{x}) = p(\mathbf{y}_j[s]|\mathbf{x})p(\mathbf{y}_k[t]|\mathbf{x}) \tag{2.13}$$

□

Under Assumption 2, we have the prior distribution

$$p(\mathcal{Y}_1 \cup \mathcal{Y}_2|\mathbf{x}) = \frac{p(\mathcal{Y}_1|\mathbf{x})p(\mathcal{Y}_2|\mathbf{x})}{p(\mathcal{Y}_1 \cap \mathcal{Y}_2|\mathbf{x})} \tag{2.14}$$

where the denominator arises to remove double-counting.

We then have the posterior distribution

$$p(\mathbf{x}|\mathcal{Y}_1 \cup \mathcal{Y}_2) = \frac{p(\mathcal{Y}_1 \cup \mathcal{Y}_2|\mathbf{x})p(\mathbf{x})}{p(\mathcal{Y}_1 \cup \mathcal{Y}_2)} \text{ by Bayes' rule}$$

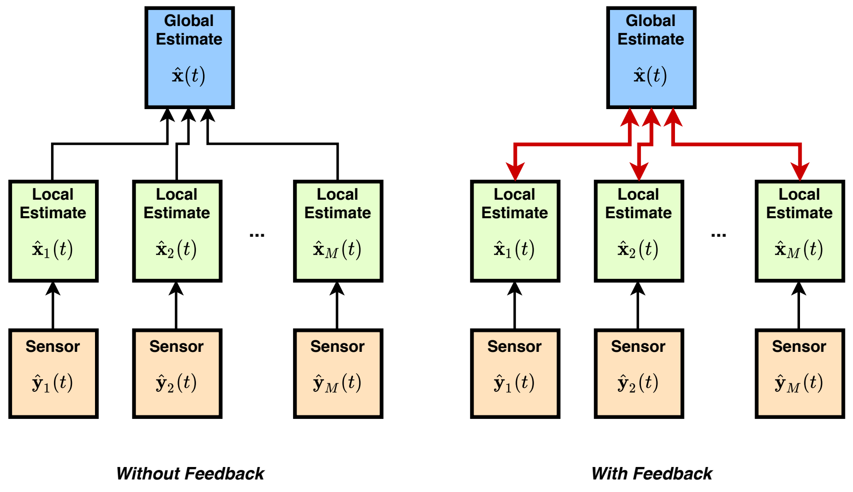

**Figure 2.1:** Caption.

$$
\begin{aligned}
&= \frac{p(\mathcal{Y}_1|\mathbf{x})p(\mathcal{Y}_2|\mathbf{x})p(\mathbf{x})}{p(\mathcal{Y}_1 \cap \mathcal{Y}_2|\mathbf{x})p(\mathcal{Y}_1 \cup \mathcal{Y}_2)} \\
&= \frac{p(\mathcal{Y}_1, \mathbf{x})p(\mathcal{Y}_2, \mathbf{x})}{p(\mathcal{Y}_1 \cap \mathcal{Y}_2, \mathbf{x})p(\mathcal{Y}_1 \cup \mathcal{Y}_2)} \text{ by multiplying } p(\mathbf{x}) \text{ top and bottom} \\
&= \frac{p(\mathbf{x}|\mathcal{Y}_1)p(\mathbf{x}|\mathcal{Y}_2)}{Cp(\mathbf{x}|\mathcal{Y}_1 \cap \mathcal{Y}_2)} \qquad (2.15)
\end{aligned}
$$

where $C$ is the normalization constant

$$
C = \frac{p(\mathcal{Y}_1 \cap \mathcal{Y}_2)p(\mathcal{Y}_1 \cup \mathcal{Y}_2)}{p(\mathcal{Y}_1)p(\mathcal{Y}_2)} \qquad (2.16)
$$

Note that $p(\mathbf{x}|\mathcal{Y}_j)$ is the local estimate of Node $j$ for $j \in \{1, 2\}$. Furthermore, $p(\mathbf{x}|\mathcal{Y}_1 \cup \mathcal{Y}_2)$ is the estimate based on their joint information, while $p(\mathbf{x}|\mathcal{Y}_1 \cap \mathcal{Y}_2)$ is the estimate based on their common information.

## 2.3 For Linear and Gaussian Distribution

We consider an environment setup similar to (1.23), but with multiple sensors instead of one. We have the following linear discrete-time state dynamics

$$
\mathbf{x}[t+1] = A[t]\mathbf{x}[t] + \mathbf{w}[t] \qquad (2.17)
$$

where $\mathbf{x}[t] \in \mathbb{R}^n$, $\mathbf{w}[t] \sim \mathcal{N}(0, Q[t])$, $A, Q[t] \in \mathbb{R}^{n\times n}$, and $t \in \mathbb{N}$. Consider a hierarchical architecture with one higher-level node in communication with $M \in \mathbb{N}$ lower-level nodes, each with measurement equation

$$
\mathbf{y}_j[t] = C_j[t]\mathbf{x}[t] + \mathbf{v}_j[t], \;\; j = 1, \cdots, M \qquad (2.18)
$$

where $\mathbf{y}_j[t] \in \mathbb{R}^{m_j}$, $\mathbf{v}_j[t] \sim \mathcal{N}(0, R_j)$, $C_j \in \mathbb{R}^{m_j \times n}$, and $R_j \in \mathbb{R}^{m_j \times m_j}$.

Define $\mathcal{Y}_j[t] := \sigma(\mathbf{y}_j[0], \cdots, \mathbf{y}_j[t])$ to be the observation set of sensor $j$ until time $t \in \mathbb{N}$. Further define $\hat{\mathbf{x}}_j[t|s] := \hat{E}[\mathbf{x}[t]|\mathcal{Y}_j[s]]$ to be the linear MMSE of $\mathbf{x}[t]$ for sensor $j$ given observation set $\mathcal{Y}_j[s]$, and let

$\hat{P}_j[t|s] := \mathrm{Cov}(\mathbf{x}[t] - \hat{\mathbf{x}}_j[t|s])$ be the covariance of error. Then the Kalman filter equations for each local estimate of sensor $j$ follow exactly as the single sensor case (1.31).

$$\hat{\mathbf{x}}_j[t|t] = A[t-1]\hat{\mathbf{x}}_j[t-1|t-1] + L_j[t]\tilde{\mathbf{y}}_j[t] = (I_n - L_j[t]C_j[t])A[t-1]\hat{\mathbf{x}}[t-1|t-1] + L_j[t]\mathbf{y}_j[t] \tag{2.19a}$$

$$\hat{P}_j[t|t] = (I_n - L_j[t]C_j[t])\hat{P}_j[t|t-1] = (I_n - L_j[t]C_j[t])(A[t-1]\hat{P}_j[t-1|t-1]A[t-1]^T + Q[t-1]) \tag{2.19b}$$

where (2.19a) simplifies by definition of the innovation process (1.25) and the Kalman gain is now defined as

$$L_j[t] := \hat{P}_j[t|t-1]C_j(C_j^T\hat{P}_j[t|t-1]C_j + R_j)^{-1} \tag{2.20}$$

For simplicity of notation, we henceforth represent $\hat{\mathbf{x}}_j[t]$ to be equivalent to $\hat{\mathbf{x}}_j[t|t]$ and $\hat{P}_j[t]$ to be equivalent to $\hat{P}_j[t|t]$.

Because there are multiple sensors, we now need to keep track of the *cross-covariance error* between two distinct sensors $j, k \in \{1, \cdots, M\}$.

$$\begin{aligned}\hat{P}_{jk}[t] &:= \mathbb{E}\left[(\mathbf{x}[t] - \hat{\mathbf{x}}_j[t])(\mathbf{x}[t] - \hat{\mathbf{x}}_k[t])^T\right] \\ &= (I_n - L_j[t]C_j[t])\left(A[t-1]\hat{P}_{jk}[t-1]A^T[t-1] + Q[t-1]\right)(I_n - L_k[t]C_k[t])^T\end{aligned} \tag{2.21}$$

In the sections that follows, we present different algorithms of accumulating local estimates from lower-level nodes in order to obtain the best fusion estimate at the upper-level processor node.

### 2.3.1 Via Dimensionality Reduction

**Idea**: Adapted from [6]. Choose a subset of scalar components from each measurement vector to transmit to the central processor. The missing components are estimated by the central processor itself, using its own past fused estimate. The total number of components transmitted across all sensors should satisfy the communication constraint.

Let $r_j[t] \leq n$ be the number of components of the vector $\hat{\mathbf{x}}_j[t]$ that sensor $j \in \{1, \cdots, M\}$ transmits to the fusion center. The other $n - r_j[t]$ components are discarded. Assume that there is a communication bandwidth constraint imposed on each channel feeding into the central processor from each sensor, encoded as a vector $\mathbf{r}[t] := [r_1[t], \cdots, r_M[t]]$.

We can represent the *reorganized state estimate (RSE)* $\hat{\mathbf{x}}_j^r[t]$ given by:

$$\hat{\mathbf{x}}_j^r[t] := H_j[t]\hat{\mathbf{x}}_j[t] \tag{2.22}$$

using diagonal matrix $H_j[t] = \mathrm{diag}(\delta_{j,1}[t], \cdots, \delta_{j,n}[t]) \in \{0,1\}^{n\times n}$, where $\delta_{j,i}[t] \in \{0,1\}$ for each $i \in \{1, \cdots, n\}$ and all $t \in \mathbb{N}$. Essentially, the indicator variable $\delta_{j,i}[t]$ is 1 if sensor $j$ chooses to transmit component $i$ of its local estimate $\hat{\mathbf{x}}_j[t]$ at time $t$.

With this setup, we are focused on two problems towards the problem of optimal fusion:

1. **Trigger Problem**: for each sensor $j$, how do we choose the $r_j[t]$ components to transmit to the central processor?

2. **Fusion Problem**: how do we aggregate all the RSEs over all sensors $j = 1, \cdots, M$ to construct the optimal estimate?

2021/11/16 Note: When the three rules of the original modular framework from OUTformation is applied to this particular setup:

- The Representation Rule can be thought of as the Kalman filtering algorithm which produces $\hat{\mathbf{x}}_j[t]$ for each sensor $j$.
- The Triggering Rule can be thought of as the optimization problem which chooses the best subset of components to transmit.
- The Fusion Problem can be thought of as the way in which all the compensated state estimates are aggregated to minimize the MSE.

For the sake of formulating both of the above problems into optimization problems, we introduce the following (redundant) mathematical notation. For each sensor $j \in \{1, \cdots, M\}$, denote $\Delta_j[t] := \binom{n}{r_j[t]}$ to be the total number of combinations of $n - r_j[t]$ 0's and $r_j[t]$ 1's along the diagonal of the matrix $H_j[t]$. Let $\mathcal{H}_j[t] := \{H_{j,h}[t], h = 1 \cdots, \Delta_j[t]\}$ be the set of all possible combinations of diagonal matrices which can be used to create the RSE. For each $h = 1, \cdots, \Delta_j[t]$, denote the indicator variable

$$\sigma_{j,h}[t] = \begin{cases} 1 & \text{if } H_{j,h}[t] \text{ is chosen at time } t \\ 0 & \text{if } H_{j,h}[t] \text{ is not chosen at time } t \end{cases} \tag{2.23}$$

and construct the correpsonding *decision variables*:

$$\xi_j[t] := \begin{bmatrix} \sigma_{j,1}[t] & \sigma_{j,2}[t] & \cdots & \sigma_{j,\Delta_j[t]}[t] \end{bmatrix}^T \in \{0,1\}^{\Delta_j[t]} \tag{2.24}$$

Note that for each $j \in \{1, \cdots, M\}$ and every time $t \in \mathbb{N}$, there can only be one value of $h \in \{1, \cdots, \Delta_j[t]\}$ such that $\sigma_{j,h}[t]$ has a value of 1. All other values are 0. Hence, $\xi_j[t]$ is equivalent to one of the basis vectors of $\mathbb{R}^{\Delta_j[t]}$, i.e. it is 0 everywhere except for a single component, where it is 1. Hence, the original diagonal matrix $H_j[t]$ chosen to represent the RSE is given by

$$H_j[t] := \operatorname{diag}(\delta_{j,1}[t], \cdots, \delta_{j,n}[t]) = \sum_{h=1}^{\Delta_j[t]} \sigma_{h,j}[t] H_{j,h}[t] \tag{2.25}$$

**Example 4.** To help clarify the notation above, we introduce the following simple example. Let dimension of the full state vector be $n = 4$ and $r_j[t] = 2$. Then $\Delta_j[t] = 6$ and the possible $H_{j,h}[t]$ matrices are

$$\begin{aligned} &H_{j,1}[t] = \operatorname{diag}(1,1,0,0), \quad H_{j,2}[t] = \operatorname{diag}(1,0,1,0), \quad H_{j,3}[t] = \operatorname{diag}(1,0,0,1) \\ &H_{j,4}[t] = \operatorname{diag}(0,1,1,0), \quad H_{j,5}[t] = \operatorname{diag}(0,1,0,1), \quad H_{j,6}[t] = \operatorname{diag}(0,0,1,1) \end{aligned}$$

If $h = 3$ is chosen at time $t$, then $\sigma_{j,3}[t] = 1$ and $\sigma_{j,h}[t] = 0$ for all $h \in \{1, \cdots, 6\}/\{3\}$. The corresponding RSE is $\hat{\mathbf{x}}_j^r[t] = \begin{bmatrix} x_1 & 0 & 0 & x_4 \end{bmatrix}^T$. □

For each $\hat{\mathbf{x}}_j^r[t]$ transmitted from sensor $j \in \{1, \cdots, M\}$ to the central processor, the central processor creates a *compensated state estimate (CSE)*, denoted $\hat{\mathbf{x}}_j^c[t]$, as follows:

$$\hat{\mathbf{x}}_j^c[t] = \hat{\mathbf{x}}_j^r[t] + (I_n - H_j[t]) A[t-1] \hat{\mathbf{x}}[t-1] \tag{2.26}$$

where $\hat{\mathbf{x}}[t-1]$ is the fused estimate at time $t-1$. We essentially leverage the assumption that the dynamics $A[t]$ is known to propagate the previous time's estimate forwards one timestep, and fill in the remaining components which are not covered by sensor $j$'s transmission. 2021/11/16 Note: Biggest issue is that the dynamics are assumed to be known, which is is typical of Kalman filtering schemes. In particular, $A[t], Q[t]$, and $R_j[t]$ for each $j = 1, \cdots, M$. We can assume $C_j[t]$, is known; this comes from a separate problem known as data association and such a problem is outside of the scope of consideration. Because the dynamics $A[t]$ are known, (2.26) can be computed. Moreover, the Kalman filter allows for local state estimates of each sensor to be made for the full vector state, even for components which are directly unobservable. In unknown dynamics case, neither of the two advantages above can be used; we can only broadcast the subset of observable components of the full state estimate to each sensor.

We can now formulate an optimization problem for each of the two problems described above:

1. **Fusion Problem**: find optimal weight matrices $\{\Omega_1^*[t], \cdots, \Omega_M^*[t]\}$ such that the mean-squared error (MSE) performance of the designed estimate

$$\hat{\mathbf{x}}^*[t] := \sum_{j=1}^{M} \Omega_j^*[t]\hat{\mathbf{x}}_j^c[t] \tag{2.27}$$

is minimized:

$$\begin{aligned} \{\hat{\mathbf{x}}^*[t], \Omega_1^*[t], \cdots, \Omega_M^*[t]\} = \underset{\{\hat{\mathbf{x}}[t], \Omega_1[t], \cdots, \Omega_M[t]\}}{\operatorname{argmin}} \; & \mathbb{E}\left[(\mathbf{x}[t] - \hat{\mathbf{x}}[t])^T(\mathbf{x}[t] - \hat{\mathbf{x}}[t])\right] \\ \text{s.t. } \hat{\mathbf{x}}[t] = & \sum_{j=1}^{M} \Omega_j[t]\hat{\mathbf{x}}_j^c[t] \\ & \sum_{j=1}^{M} \Omega_j[t] = I_n \end{aligned} \tag{2.28}$$

2021/11/16 Note: For the sake of simplicity, ignore the fusion problem; use equal weighting across all sensors. Also assuming for the sake of simplicity that there is no memory in the sensor nodes and in the central processor node, which implies there is no need to optimize weights over previous values in time.

2. **Trigger Problem**: based on the optimal fusion estimate $\hat{\mathbf{x}}[t]$ determined from the Fusion Problem above, choose a group of optimal decision variables $\{\xi_1^*[t], \cdots, \xi_M^*[t]\}$ such that the MSE of the optimal fusion estimate is minimal:

$$\begin{aligned} \{\xi_1^*[t], \cdots, \xi_M^*[t]\} = \operatorname{argmin} \mathbb{E}\left[(\mathbf{x}[t] - \hat{\mathbf{x}}^*[t])^T(\mathbf{x}[t] - \hat{\mathbf{x}}^*[t])\right] \\ \text{s.t. } \sum_{j=1}^{M} r_j[t] = r \\ H_j[t] = \operatorname{diag}(\delta_{j,1}[t], \cdots, \delta_{j,n}[t]) \; \forall j = 1, \cdots, M \\ \sum_{i=1}^{n} \delta_{j,i}[t] = r_j[t] \; \forall j = 1, \cdots, M \end{aligned} \tag{2.29}$$

Note that the optimization is done over the variables $\delta_{j,1}[t], \cdots, \delta_{j,n}[t]$, related to $\xi_1[t], \cdots, \xi_M[t]$ through the relationship detailed in the notation description above.

2021/11/16 Note: Biggest issue with this approach is that the optimization problems for both the Fusion Problem and the Trigger Problem rely on knowing the value of the true $\mathbf{x}[t]$, i.e. objective function $\mathbb{E}\left[(\mathbf{x}[t]-\hat{\mathbf{x}}^*[t])^T(\mathbf{x}[t]-\hat{\mathbf{x}}^*[t])\right]$. One way to resolve this for the OUTformation approach: the Trigger Problem is solved locally for each sensor nose $j$ and incorporates the broadcast vector $\hat{\mathbf{x}}[t]$ obtained from the central processor. For all $j = 1, \cdots, M$:

$$\begin{aligned}\{\delta^*_{j,1}[t], \cdots, \delta^*_{j,n}[t]\} = \operatorname{argmin} \mathbb{E}\left[(\hat{\mathbf{x}}[t]-\hat{\mathbf{x}}^*_j[t])^T(\hat{\mathbf{x}}[t]-\hat{\mathbf{x}}^*_j[t])\right] \\ \text{s.t.} \sum_{i=1}^{n} \delta_{j,i}[t] = r_j[t] \\ H_j[t] = \operatorname{diag}(\delta_{j,1}[t], \cdots, \delta_{j,n}[t])\end{aligned} \tag{2.30}$$

(Not quite right because the 0 components in $\hat{\mathbf{x}}^*_j[t]$ should still be compensated.)

$$P[t|t] := \mathbb{E}\left[(\mathbf{x}[t]-\hat{\mathbf{x}}[t])^T(\mathbf{x}[t]-\hat{\mathbf{x}}[t])\right] \tag{2.31}$$

Note that both problems require knowledge of the true state $\mathbf{x}[t]$ to compute, since the performance objectives are dependent on it. The two problems are alternatingly solved and repeated for each timestep $t = 1, \cdots, T$, for some designated terminal time $T \in \mathbb{N}$. We now present the pseudocode used to solve each optimization problem.

**The Fusion Problem** Define error variables for the CSE $\mathbf{e}_j[t] := \mathbf{x}[t] - \hat{\mathbf{x}}^c_j[t]$ and error covariance matrix

$$\Sigma[t] := \mathbb{E}\left[\begin{bmatrix} e_1[t] \\ e_2[t] \\ \vdots \\ e_M[t]\end{bmatrix}\begin{bmatrix} e_1^T[t] & e_2^T[t] & \cdots & e_M^T[t]\end{bmatrix}\right] \tag{2.32}$$

Then the optimal matrix weights $\{\Omega^*_1[t], \cdots, \Omega^*_M[t]\}$ can be computed straightforwardly using least mean squares:

$$\begin{bmatrix}\Omega^*_1[t] & \Omega^*_2[t] & \cdots & \Omega^*_M[t]\end{bmatrix} = \left(I^T_{n,M}\Sigma^{-1}[t]I_{n,M}\right)^{-1} I^T_{n,M}\Sigma^{-1}[t] \tag{2.33}$$

where $I_{n,M} := [I_n, \cdots I_n]^T \in \{0,1\}^{Mn\times n}$. Furthermore, the error covariance $\hat{P}[t] := \mathbb{E}[(\mathbf{x}[t]-\hat{\mathbf{x}}[t])(\mathbf{x}[t]-\hat{\mathbf{x}}[t])^T]$ of the fused central processor estimate can be computed as follows:

$$\hat{P}[t] = \left(I^T_{n,M}\hat{P}^{-1}[t]I_{n,M}\right)^{-1} \tag{2.34}$$

As is common with computation involving large-dimensional matrices, the inverses in (2.33) and (2.34) cannot be computed directly. Thus, the optimization problem used to solve the Fusion Problem boils down to standard matrix approximation algorithms used to bypass computation of the inverses in (2.33) and (2.34).

**The Trigger Problem** Define the set of all feasible transmissions to the central processor, i.e. the set of all bandwidth consumption profiles in which every sensor transmits such that the total bandwidth constraint $r \in \mathbb{N}$ is satisfied:

$$\mathcal{R}_r[t] := \left\{ \begin{bmatrix} r_1[t] \\ r_2[t] \\ \vdots \\ r_M[t] \end{bmatrix} \in \mathbb{N}_+^M \,\middle|\, \sum_{j=1}^{M} r_j[t] = r, \;\; 1 \le r_j[t] \le n \right\} \tag{2.35}$$

Denote $\eta_j[t] := \left[\delta_{j,1}[t], \cdots, \delta_{j,n}[t]\right]^T \in \mathbb{R}^n$ for all $j = 1, \cdots, M$. That is, $\eta_j[t]$ represents the diagonal elements of $H_j[t]$ put in vector form. Define matrix $U_j[t] \in \mathbb{R}^{n \times \Delta_j[t]}$ such that $\eta_j[t]$ and let the decision variables satisfy

$$U_j[t]\xi_j[t] = \eta_j[t] \quad \forall j = 1, \cdots, M \tag{2.36}$$

Note that

$$\sum_{\ell=1}^{n} \eta_{j,\ell}[t] = r_j[t] \quad \Longrightarrow \quad \sum_{j=1}^{M}\sum_{\ell=1}^{n} \eta_{j,\ell} = \sum_{j=1}^{M} r_j[t] = r \tag{2.37}$$

**Example 5.** We continue from Example 4 to show that a $U_j[t]$ which satisfies (2.36) always exists. Suppose $\eta_j[t] = [1, 0, 0, 1]^T \in \mathbb{R}^4$, corresponding to $h = 3$. Then we have the $\xi_j[t] = [0, 0, 1, 0, 0, 0]^T \in \mathbb{R}^6$. One such $U_j[t]$ which satisfies (2.36) is given by

$$\begin{bmatrix} * & * & 1 & * & * & * \\ * & * & 0 & * & * & * \\ * & * & 0 & * & * & * \\ * & * & 1 & * & * & * \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{2.38}$$

where $*$ can be replaced by any real value. □

As mentioned in the discussion of the Fusion Problem, we cannot compute the inverses in (2.33) and (2.34) directly. Hence, we require an approximation. Using the variables defined above, we can solve (2.29) using the following approximate optimization problem:

$$\begin{aligned} \{\eta_1^*[t], \cdots, \eta_M^*[t]\} = \underset{\{\eta_1[t], \cdots, \eta_M[t]\}}{\operatorname{argmin}} \quad & \operatorname{Trace}(\Sigma[t]) \\ \text{s.t.} \; & \sum_{j=1}^{M} r_j[t] = r \;\forall j = 1, \cdots, M \\ & \mathbb{1}_n \eta_j[t] = r_j[t] \;\forall 1 \le r_j[t] \le n, \;\; j = 1, \cdots, M \\ & \delta_{j,i}[t] \in \{0, 1\} \;\forall i = 1, \cdots, n, \;\; j = 1, \cdots, M \end{aligned} \tag{2.39}$$

where $\mathbb{1}_n$ is a vector of all ones with dimension $n$.

We can further divide (2.39) into two steps.

1. for the $k$th element in the set $\mathcal{R}_r[t]$ from (2.35), where $k = 1, \cdots, |\mathcal{R}_r[t]|$, solve (2.39):

$$f_k\left(\eta_1^k[t], \cdots, \eta_M^k[t]\right) := \underset{\{\eta_1[t], \cdots, \eta_M[t]\}}{\operatorname{argmin}} \quad \operatorname{Trace}(\Sigma[t]) \tag{2.40}$$

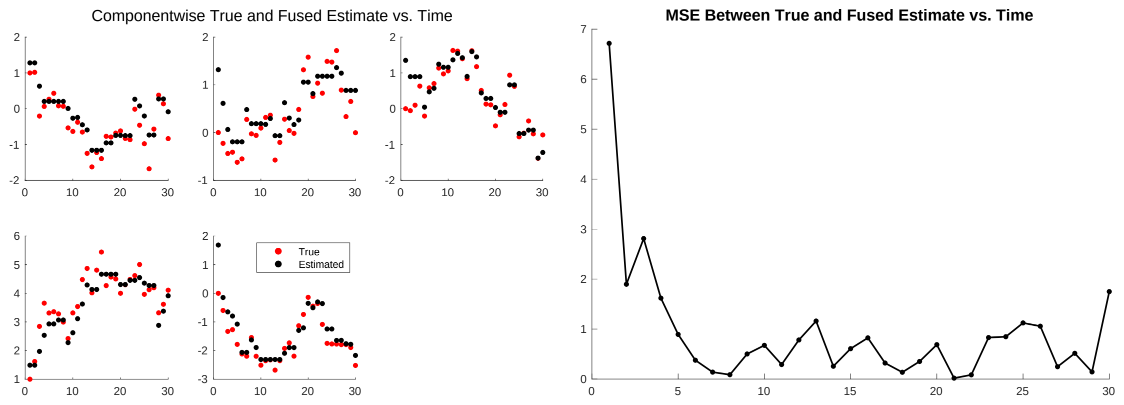

**Figure 2.2:** Preliminary figures for a simple example demonstrating the dimensionality reduction multi-sensor fusion approach.

$$\begin{aligned} \text{s.t. } & [r_1^k, \cdots, r_M^k]^T \in \mathcal{R}_r \\ & \mathbb{1}_n \eta_j[t] = r_j^k[t] \; \forall 1 \le r_j^k[t] \le n, \; j = 1, \cdots, M \\ & \delta_{j,i}[t] \in \{0,1\} \; \forall i = 1, \cdots, n, \; j = 1, \cdots, M \end{aligned}$$

2. after obtaining all $|\mathcal{R}_r[t]|$ optimal sets by solving (2.40) for each $k = 1, \cdots, |\mathcal{R}_r[t]|$, take the minimum value:

$$\{\eta_1^*[t], \cdots, \eta_M^*[t]\} = \text{argmin}\left\{ f_1\left(\eta_1^k[t], \cdots, \eta_M^k[t]\right), \cdots, f_{|\mathcal{R}_r[t]|}\left(\eta_1^k[t], \cdots, \eta_M^k[t]\right)\right\} \tag{2.41}$$

2021/11/16 Note: Additional distinctions of this Kalman filter multi-sensoring approach to the original modular framework designed for OUTformation:

- Need to include a separate bandwidth constraint on the downlink broadcast inside Trigger Problem (2.30), which should be related to the power constraint of receiving signals for each sensor.
- No communication delays. Related to this, no periodic asynchronous timing of different transmissions, i.e. only having synchronous computation of estimates for both sensors and central processor at each timestep.

### 2.3.2 Via Feedback Loops

Notation change. $\Sigma \to \hat{P}$ and $P \to ?$

For easier combination of the $M$ separate estimates, it is often useful to represent to Kalman filter using its *information form* [7]. Define the *information matrix* and the *information vector* to be, respectively:

$$P_j[t|s] := \Sigma_j^{-1}[t|s], \quad \hat{\mathbf{z}}_j[t|s] := \Sigma_j^{-1}[t|s]\hat{\mathbf{x}}_j[t|s] \tag{2.42}$$

**Lemma 2** (Matrix Inversion Lemma)**.** For matrices $A \in \mathbb{R}^{n\times n}, D \in \mathbb{R}^{k\times k}, U \in \mathbb{R}^{n\times k}$, and $V \in \mathbb{R}^{k\times n}$

$$(A + UDV)^{-1} = A^{-1} - A^{-1}U(D^{-1} + VA^{-1}U)^{-1}VA^{-1}$$

Using Lemma 2, the local update equations transform to

$$P_j[t|t] = P_j[t|t-1] + C_j^T R_j^{-1} C_j, \quad \hat{\mathbf{z}}_j[t|t] = \hat{\mathbf{z}}_j[t|t-1] + C_j^T R_j^{-1}(\mathbf{y}_j[t] - C_j \hat{\mathbf{z}}_j[t|t-1]) \tag{2.43}$$

In combination, the overall estimate is simply a sum over all the independent local estimates of the lower-level nodes:

$$P[t|t] = P[t|t-1] + \sum_{j=1}^{M} C_j^T R_j^{-1} C_j, \quad \hat{\mathbf{z}}[t|t] = \hat{\mathbf{z}}[t|t-1] + \sum_{j=1}^{M} C_j^T R_j^{-1}(\mathbf{y}_j[t] - C_j \hat{\mathbf{z}}_j[t|t-1]) \tag{2.44}$$

where $P[t|s]$ represents the covariance of error of the higher-level node's overall estimate.

Using (2.43), the fusion equations (2.44) simplify as

$$P[t|t] = \sum_{j=1}^{M} (P_j[t|t] - P_j[t|t-1]) + P[t|t-1], \quad \hat{\mathbf{z}}[t|t] = \sum_{j=1}^{M} (\hat{\mathbf{z}}_j[t|t] - \hat{\mathbf{z}}_j[t|t-1]) + \hat{\mathbf{z}}_j[t|t] \tag{2.45}$$

When the $M$ lower-level estimators and the higher-level estimator share the same prior information on the state $\mathbf{x}(t)$, the fusion equations (2.44) change as follows

$$P[t|t] = \sum_{j=1}^{M} P_j[t|t] - (M-1)P[t|t-1], \quad \hat{\mathbf{z}}[t|t] = \sum_{j=1}^{M} \hat{\mathbf{z}}_j[t|t] - (M-1)\hat{\mathbf{z}}[t|t-1] \tag{2.46}$$

which arises since $\hat{\mathbf{z}}_j[t|t-1] = \hat{\mathbf{z}}[t|t-1]$ and $P_j[t|t-1] = P[t|t-1]$ for all $j = 1, \cdots, M$ under the common prior information assumption.

**Remark 5.** Maintaining the same prior information on the state $\mathbf{x}(t)$ across all $M$ lower-level nodes and the higher-level node can be achieved by adding a feedback communication channel from the higher-level node to the lower-level nodes. The paper [4] distinguishes between the case of shared and unshared prior information as the "feedback case" and a "non-feedback case", respectively. See Figure 2.1 for a visualization of the two architectures. $\square$

## 2.4 Comparisons between Feedback and Non-Feedback Architectures

### 2.4.1 Modular Framework

We focus on distributed data-fusion methods with applications to state estimation or reference-tracking. Many distributed data fusion algorithms with multiple layers of processors and sensors have been proposed in the past few decades (see [5, 4] for surveys). For the purposes of our paper, it suffices to focus on the simplest setup where a single central processor communicates with multiple independent external sensors.

Denote the state of the environment by vector $\mathbf{x}(t) \in \mathbb{R}^n$. For each component $i = 1, \cdots, n$, the dynamics vary according to the equation

$$dx_i = f_i(t, \mathbf{x}(t), \mathbf{d}(t)) \tag{2.47}$$

Here, function $f_i : \mathbb{R}^+ \times \mathbb{R}^n \times \mathbb{R}^d \to \mathbb{R}$ takes as inputs time $t$, state $\mathbf{x}(t)$, and external noise vector $\mathbf{d}(t) \in \mathbb{R}^d$. The noise can be modeled according to any stochastic process, e.g., Gaussian white noise or Poisson shot noise [8].

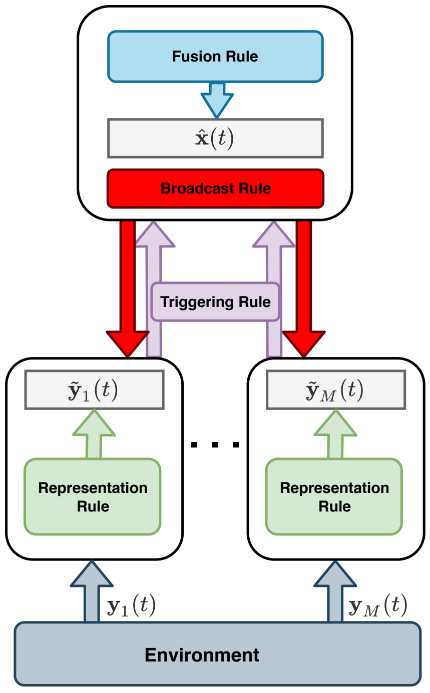

**Figure 2.3:** A modular representation of an OUTformation architecture for distributed state estimation under unknown environments and communication delay constraints, using a single central processor and a network of $M$ external sensors. The representation rule is highlighted in green, the triggering rule in purple, and the fusion rule in blue. The downlink red arrows indicate the broadcast capability of the central processor, which is not present for INformation architectures.

We use a distributed network of $M \in \mathbb{N}$ sensors to make observations about the state of the environment. For each $j \in \{1, \cdots, M\}$, sensor $j$ makes a noisy measurement of some subset of $m_j < n$ components of the state vector according to its measurement equation. For simplicity, we employ a linear measurement equation perturbed by additive, white Gaussian noise, written as

$$\mathbf{y}_j(t) = C_j \mathbf{x}(t) + \mathbf{v}_j(t) \tag{2.48}$$

Here, $C_j \in \{0, 1\}^{m_j \times n}$, each row contains exactly one 1, and each column contains at most one 1. The vector $\mathbf{v}_j(t) \sim \mathcal{N}(\mathbf{0}, \Sigma_j)$ represents white Gaussian noise.

The central processor receives data that is transmitted to it by the external sensors. The sensors only communicate with the central processor, never among themselves. In the *distributed estimation problem*, the central processor aggregates the transmissions of the sensors' local measurements $\{\mathbf{y}_1(t), \cdots, \mathbf{y}_M(t)\}$ to create an estimate $\hat{\mathbf{x}}(t)$ of the environment's true state $\mathbf{x}(t)$. Given the problem setup described above, the traditional INformation approach and the proposed OUTformation approach may be distinguished in the following way.

**Definition 4** (INformation and OUTformation Approaches)**.** In an *INformation approach*, each sensor's local measurement $\tilde{\mathbf{y}}(t)$ is constructed based only on local knowledge, e.g., local observations of the environment state. In an *OUTformation approach* to distributed estimation is any approach which constructs $\hat{\mathbf{x}}(t)$ based on both local knowledge and the central processor's knowledge, which is broadcast to the sensors. A model of the OUTformation approach is depicted in Figure 2.3, while a model of the INformation approach can be depicted by removing the red broadcast arrows in Figure 2.3. □

We use the terminology INformation versus OUTformation because the presence or absence of feedback communication (channels which push data "outward" from the central processors to the sensors) is what differentiates the two architectures. Furthermore, OUTformation architectures still include feedforward communication (data transmitted "inward") from the sensors to the central processor.

**Definition 5** (Uplink and Downlink)**.** We say that data is transmitted in the *uplink* direction if it is sent from a sensor to the central processor. Likewise, we say that data is broadcast in the *downlink* direction if it is sent from the central processor to a sensor. □

Before describing the key functions of the modular framework, we first define a few parameters and make the following operation assumptions.

**Notation 1** (Sampling Period and Communication Delays)**.** The following notation describes the parameters used for the system in Figure 2.3:

- $T_j^{(s)}$: the sampling period of sensor $j$, sensor $j$ collects one observation every $T_j^{(s)}$ seconds.
- $\Delta t^{(s)}$: the time delay incurred in transmitting a single packet from any sensor to the central processor.
- $\Delta t^{(b)}$: the time delay incurred in broadcasting a single packet from the central processor to the sensor.

**Assumption 3** (Fast Sensor Sampling Rate)**.** For each sensor $j \in \{1, \cdots, M\}$, the sampling rate $T_j^{(s)}$ is fast enough so that there is always a fresh observation available when it is time for a sensor to transmit, i.e. both $T_j^{(I)}$ and $T_j^{(O)}$ are divisible by $T_j^{(s)}$. □

**Assumption 4** (Power Expenditure per Component)**.** The central processor is connected to an unlimited power source; the sensors operate on limited sources, such as batteries. □

Throughout the remainder of the paper, other parameters corresponding to the framework Figure 2.3 are assigned the superscript $(I)$ for INformation architectures, and the superscript $(O)$ for OUTformation architectures. We further denote the time series of vectors $\{\mathbf{y}(t_1), \cdots, \mathbf{y}(t_2)\} \triangleq \mathbf{y}(t_1 : t_2)$ for any $0 \le t_1 < t_2$. The performance metrics of consideration for both types of architectures are as follows. First, the *average mean-squared error (MSE)* metric for each component $i \in \{1, \cdots, n\}$ of the environment state vector over time $[0, T_{\text{sim}})$ is given by

$$\frac{1}{T_{\text{sim}}} \sum_{t=0}^{T_{\text{sim}}} \left| x_i(t) - \hat{x}_i^{(\chi)}(t) \right|^2, \quad \chi \in \{I, O\} \tag{2.49}$$

Second, we consider the *total power expenditure* of an algorithm.

**Definition 6** (Power Expenditure of a Packet)**.** Denote $P_U$ to be the amount of power used by a sensor to transmit a packet in the uplink direction. Similarly, denote $P_D$ to be the amount of power used by a sensor to receive a packet in the downlink direction. □

### 2.4.2 Setting 1: Periodic Sensor Transmissions

Before describing the key functions of the modular framework, we first define a few parameters and make the following operation assumptions.

**Definition 7** (Transmission Periods)**.** We make the following definitions for all components of Figure 2.3:

- $T_j^{(I)}$: the baseline period of transmission of sensor $j$ using the INformation strategy. Sensor $j$ sends its packets, if there are any, to the central processor every $T_j^{(I)}$ seconds using the INformation strategy.
- $T_j^{(O)}$: the baseline period of transmission of sensor $j$ using the OUTformation strategy.

□

Because a broadcast at time $t$ transmits the full vector $\mathbf{x}^{(b)}(t) \in \mathbb{R}^n$, the total broadcast delay is given by $n\Delta t^{(b)}$. Both the central processor and the external sensors have finite memory. Namely, each sensor saves only a fixed number of past observations of the environment, and the central processor saves a fixed number of packets it receives from its queue.

Each sensor $j$ saves two quantities in its memory: 1) the previous transmission it made to the central processor, and 2) the past $H_j^{(s)}$ observations $\mathbf{y}_j(t)$ in a table of observations, where $H_j^{(s)} \in \mathbb{N}$. At time $t \geq 0$, the observation table (shown in green in Figure 2.3) is given by $\mathcal{Y}_j(\lfloor t/T_j^{(s)} \rfloor)$, where $\mathcal{Y}_j(c) := \mathbf{y}_j(\max\{0, c - H_j^{(s)} + 1\}T_j^{(s)} : cT_j^{(s)})$ for any $c \in \mathbb{N}$. For sensor $j \in \{1, \cdots, M\}$, the oldest observation is dropped from the observation table when a new observation is made at time $(c+1)T_j^{(s)}$. That is,

$$\begin{aligned} \mathcal{Y}_j(c) &:= \mathbf{y}_j(\max\{0, c - H_j^{(s)} + 1\}T_j^{(s)} : cT_j^{(s)}) \\ &\downarrow \\ \mathcal{Y}_j(c+1) &:= \mathbf{y}_j(\max\{0, c - H_j^{(s)} + 2\}T_j^{(s)} : (c+1)T_j^{(s)}) \end{aligned} \tag{2.50}$$

Hence, at any time $t \geq 0$, there are exactly $\min\{H_j^{(s)}, \lfloor t/T_j^{(s)} \rfloor\}$ entries in sensor $j$'s observation table.

A *representation rule* $\alpha_j : \mathcal{S}_j \to \mathbb{R}^{m_j}$ (also shown in green) is used by each sensor $j$ to construct a *representative measurement* $\tilde{\mathbf{y}}_j(t) \in \mathbb{R}^{m_j}$. Here, $\mathcal{S}_j$ denotes the set of all finite subsets of $\mathbb{R}^{m_j}$ with cardinality at most $H_j^{(s)}$. The representation rule is a function of the past $H_j^{(s)}$ observations collected from the environment until time $t$.

$$\tilde{\mathbf{y}}_j(t) = \alpha_j(\mathcal{Y}_j(\lfloor t/T_j^{(s)} \rfloor)) \tag{2.51}$$

The representative measurement is what each sensor transmits to the central processor in order for the processor to construct its estimate of the full state. The representative measurement can also be viewed as the sensor's local estimate of the partial environment state.

Sensor $j$ now needs to transmit its representative measurement $\tilde{\mathbf{y}}_j(t)$ to the central processor. For $k = 1, \cdots, m_j$, each scalar component $\tilde{y}_{jk}(t)$ of the representative measurement vector $\tilde{\mathbf{y}}_j(t)$ is encoded into *packets* via an *encoding function* $\phi : \mathbb{R}^3 \times \mathbb{R} \to \mathcal{D}$, where $\mathcal{D}$ is the space of encoded values. Here, $\phi$ takes as arguments the vector of identification tags $\nu_{jk} := (j, i_k, k) \in \mathbb{R}^3$ of the value $\tilde{y}_{jk}(t)$, and the value $\tilde{y}_{jk}(t) \in \mathbb{R}$ itself. The index $i_k \in \{1, \cdots, n\}$ refers to the component index with respect to the true state, i.e. the column position of the 1 in the $k$th row of the $C_j$ matrix from the measurement equation (2.48). The central processor decodes each received packet using the corresponding *decoding function* $\phi^{-1} : \mathcal{D} \to \mathbb{R}^3 \times \mathbb{R}$. We assume that $\phi$ and $\phi^{-1}$ satisfy the following assumption.

**Assumption 5** (Perfect Reconstruction from Packets)**.** The coding functions $\phi$ and $\phi^{-1}$ are such that the data is perfectly reconstructed. That is, for each packet $\phi(\nu_{jk}, \tilde{y}_{jk})$, the central processor uses $\phi^{-1}$ so that $\nu_{jk}$ and $\tilde{y}_{jk}$ are known perfectly without quantization effects. □

Each sensor $j$ uses a *triggering rule* (shown in violet in Figure 2.3) to determine when and which packets of the representative measurement it should send to the central processor. The central processor makes a *broadcast estimate* $\hat{\mathbf{x}}^{(b)}(t) \in \mathbb{R}^n$ (shown in red in Figure 2.3) which is equal to its estimate of the environment state. All units use the encoder $\phi$ to encode each component of the estimate vector into packets, and each sensor uses the decoder $\phi^{-1}$ to obtain the original broadcast estimate.

Because the triggering rule is dependent on the availability of the broadcast, we distinguish between a feedback triggering rule and an nonfeedback triggering rule. Let $\gamma_j^{(I)} : \mathbb{R}^3 \times \mathbb{R} \times \mathbb{R} \times \mathcal{S}_j \times \mathcal{P}_j^{(I)} \to \{0,1\}$ be the function for the triggering rule in a nonfeedback architecture, and let $\gamma_j^{(O)} : \mathbb{R}^n \times \mathbb{R}^3 \times \mathbb{R} \times \mathcal{S}_j \times \mathcal{P}_j^{(O)} \to \{0,1\}$ be that for a feedback architecture. Denote $\chi$ to be a placeholder variable such that $\chi = I$ for a nonfeedback architecture, and $\chi = O$ for a feedback architecture. Here, $P_j^{(\chi)} \in \mathcal{P}_j^{(\chi)}$ is the parameter set used for the triggering rule of architecture $\chi$. Under architecture $\chi \in \{I, O\}$, transmissions from sensor $j$ to the central processor are only made at times which are constant multiples of $T_j^{(\chi)}$, which are defined in Definition 7.

Thus, under architecture $\chi$, the set of packets transmitted from sensor $j$ to the central processor's queue at time $cT_j^{(\chi)}$, for some $c \in \mathbb{N}$, is given by

$$\mathcal{G}_j^{(\chi)}(c) := \left\{ \phi\left(\nu_{jk}, \tilde{y}_{jk}(cT_j^{(\chi)})\right), k \in \{1, \cdots, m_j\} : \gamma_{jk}^{(\chi)}(c) = 1 \right\} \tag{2.52}$$

where we write as shorthand

$$\gamma_{jk}^{(I)}(c) \triangleq \gamma_j^{(I)}\left(\nu_{jk}, \tilde{y}_{jk}(\underline{c}_{jk}T_j^{(I)}), \tilde{y}_{jk}(cT_j^{(I)}), \mathcal{Y}_j\left(\left\lfloor \frac{t}{T_j^{(s)}} \right\rfloor\right), P_j^{(I)}\right) \tag{2.53}$$

and likewise

$$\gamma_{jk}^{(O)}(c) \triangleq \gamma_j^{(O)}\left(\hat{\mathbf{x}}^{(b)}(t), \nu_{jk}, \tilde{y}_{jk}(\underline{c}_{jk}^{(O)}T_j^{(O)}), \tilde{y}_{jk}(cT_j^{(O)}), \mathcal{Y}_j\left(\left\lfloor \frac{t}{T_j^{(s)}} \right\rfloor\right), P_j^{(O)}\right) \tag{2.54}$$

In either (2.53) or (2.54), we denote

$$\underline{c}_{jk}^{(\chi)} \triangleq \max\{c^* \in \mathbb{N}, c^* < c : \gamma_{jk}^{(\chi)}(c^*) = 1\} \tag{2.55}$$

Let $U_{j,c}^{(\chi)} \triangleq |\mathcal{G}_j^{(\chi)}(c)| \leq m_j$ be the total number of packets transmitted uplink by sensor $j$ at time $cT_j^{(\chi)}$ under architecture $\chi \in \{I, O\}$. Then the central processor receives sensor $j$'s transmission at a delayed time of $cT_j^{(\chi)} + U_{j,c}^{(\chi)}\Delta t^{(s)}$ under architecture $\chi$. As it receives delayed transmissions from all of the sensors, the central processor saves the past $H_c \in \mathbb{N}$ received values in its memory, which we refer to as its *fusion table* (shown in light blue in Figure 2.3). The fusion table is represented by the matrix $Z^{(\chi)}(t) \in \mathbb{R}^{H_c \times n}$, constructed as follows for any time $t \geq 0$. As defined in Definition 7, the fusion table takes $\Delta t_c$ timesteps to service a single packet $\phi(\nu_{jk}, \tilde{y}_{jk})$ from its queue of received, unprocessed representative measurements; this includes the time to decode the packet using $\phi^{-1}$.

When packet $\phi(\nu_{jk}, \tilde{y}_{jk})$ is processed at some time $t > 0$, the $i_k$th column $Z_{\cdot,i_k}^{(\chi)}$ of $Z^{(\chi)}$ is updated as:

$$\begin{gathered} Z_{\cdot,i_k}^{(\chi)}(t) := \{Z_{1,i_k}^{(\chi)}(t), \cdots, Z_{H_c,i_k}^{(\chi)}(t)\} \\ \downarrow \\ Z_{\cdot,i_k}^{(\chi)}(t + \Delta t_c) := \{\tilde{y}_{jk}, Z_{1,i_k}^{(\chi)}(t), \cdots, Z_{H_c-1,i_k}^{(\chi)}(t)\} \end{gathered} \tag{2.56}$$

where $Z^{(\chi)}_{\cdot,i_k}$ denotes the $i_k$th column of $Z^{(\chi)}$, and $Z^{(\chi)}_{h,i_k}$ is the $(h, i_k)$th entry of $Z^{(\chi)}$. We refer to an entry $Z^{(\chi)}_{h,i}(t)$ of $Z^{(\chi)}(t)$ as "empty" if there is no packet value occupying it. Otherwise, the entry is said to be "nonempty". Note that at time $t = 0$, no packets will have arrived in the queue, and so all entries of $Z^{(\chi)}(0)$ are empty.

Let $\hat{\mathbf{x}}^{(\chi)}(t)$ denote the state estimate computed by using the INformation approach ($\chi = I$) or the OUTformation approach ($\chi = O$) at time $t$. Then $\hat{\mathbf{x}}^{(\chi)}(t) \in \mathbb{R}^n$ is created by aggregating the columns of its fusion table $Z^{(\chi)}(t)$ according to a *fusion rule* $\psi \triangleq (\psi_1, \cdots, \psi_n)$ where $\psi_i : \mathbb{R}^{1\times H_c} \to \mathbb{R}$ (also shown in light blue).

$$\hat{\mathbf{x}}^{(\chi)}_i(t) = \psi_i(Z^{(\chi)}_{\cdot,i}(t)), \quad i = 1, \cdots, n \tag{2.57}$$

Broadcast estimates $\hat{\mathbf{x}}^{(\chi)}(t)$ are sent to the sensors according to a *broadcast rule* $\beta : \mathbb{R}^+ \times \mathbb{R}^{H_c\times n} \to \{0, 1\}$. The function $\beta$ takes as input the current time and the state of the central processor and outputs a binary flag which indicates whether the current estimate $\hat{\mathbf{x}}(t)$ should be broadcast to the sensors or not. For example, under a periodic broadcasting scheme with period $T^{(b)} > 0$, $\beta(t, Z^{(O)}(t)) = 1$ if $t$ is divisible by $T^{(b)}$, 0 otherwise.

In this analysis and demonstration only, we assume the communication structure described in Section 2.4.2.

**Assumption 6** (Memoryless Environment)**.** For each state component $i \in \{1, \cdots, n\}$, the evolution of the true state $x_i(t)$ over time behaves in the following way. In each inter-transmission period $[cT^{(\chi)}_j, (c+1)T^{(\chi)}_j)$ for sensor $j \in \{1, \cdots, M\}$, architecture $\chi \in \{I, O\}$, and any $c \in \mathbb{N}$, $|x_i(cT^{(\chi)}_j) - x_i((c+1)T^{(\chi)}_j -)| > \rho$ with known probability $q^{(\chi)}_{ij}(\rho)$, where $\rho = \theta$ from (2.78) if $\chi = I$ and $\rho = \epsilon$ from (2.79) if $\chi = O$. We henceforth refer to such a phenomenon as a *$\rho$-change* in the environment's $i$th component. □

For both the feedback and nonfeedback architectures considered in this section, we make the following assumptions.

**Assumption 7** (No-Memory Components)**.** Each sensor $j \in \{1, \cdots, M\}$ only stores the most recent transmission, i.e., $H^{(s)}_j = 1$. The central processor does not keep track of previous received estimates, i.e., $H_c = 1$. □

**Assumption 8** (No Self-Overlapping Communications)**.** For any feedback architecture considered in this section, the following property holds:

$$m_j \Delta t^{(s)} + n\Delta t^{(b)} \le T^{(O)}_j \quad \forall j \in \{1, \cdots, M\} \tag{2.58}$$

That is, there is enough time between consecutive transmissions at times $cT^{(O)}_j$ and $(c + 1)T^{(O)}_j$ of sensor $j$ for the central processor to receive the transmission made at $cT^{(O)}_j$ and for the sensors to receive the corresponding broadcast if the central processor decides to make one. □

Next, we specify the representation, triggering, and fusion rules from Figure 2.3 in the following ways. Under Assumption 7 and Assumption 3, the representation rule is such that each sensor's representative measurement is simply the most recent observation:

$$\tilde{\mathbf{y}}_j(t) = \alpha_j(\mathbf{y}_j(t)) \triangleq \mathbf{y}_j(t), \;\; j \in \{1, \cdots, M\} \tag{2.59}$$

and the fusion rule replaces components of the central processor's current estimate based on what it receives from the sensors at the current time:

$$\hat{\mathbf{x}}_{i_k}^{(\chi)}(t) = Z_{1,i_k}^{(\chi)}(t) \triangleq \tilde{y}_{j'k}(\bar{t}_k^{(\chi)}) \tag{2.60}$$

where

$$\begin{aligned} j' &\triangleq \text{argmax}(\bar{t}_k^{(\chi)}) \\ \bar{t}_k^{(\chi)} &\triangleq \max_{j\in\{1,\cdots,M\}} \left\{ s_j^{(\chi)}(c_j), c_j \in \mathcal{C}_{jk}^{(\chi)} : s_j^{(\chi)}(c_j) - c_j T_j^{(\chi)} + U_{j,c_j}^{(\chi)} \Delta t^{(s)}, s_j(c_j) \leq t \right\} \\ \mathcal{C}_{jk}^{(\chi)} &\triangleq \{c_j \in \mathbb{N} : \gamma_{jk}^{(\chi)}(c_j) = 1\} \end{aligned} \tag{2.61}$$

We specifically choose the triggering rule for a nonfeedback architecture to be the following at time $cT_j^{(I)}, c \in \mathbb{N}$:

$$\gamma_{jk}^{(I)}(c) = \mathbb{1}\left\{ |y_{jk}(\underline{c}_{jk}^{(I)} T_j^{(I)}) - y_{jk}(cT_j^{(I)})| > \theta \right\} \tag{2.62}$$

Here, $T_j^{(I)}$ is defined in Definition 7, and the threshold parameter $\theta > 0$ is chosen by design. Essentially, for each component $k \in \{1, \cdots, m_j\}$, (2.78) compares the current representative measurement at time $cT_j^{(I)}$ to the previous transmission of packet $k$ at time $\underline{c}_{jk}(T_j^{(I)})$, and transmits if the difference is deemed "large enough" ($> \theta$). We henceforth refer to (2.78) as the *absolute-difference nonfeedback triggering* strategy.

For feedback architectures, we consider a similar thresholding criterion, with a key difference being the availability of the broadcast of the central processor's estimate $\hat{\mathbf{x}}^{(b)}$. However, if the latest broadcast contains information that is more outdated than the previous transmission, we revert back to the absolute-difference nonfeedback triggering strategy. That is, for $cT_j^{(O)}, c \in \mathbb{N}$,

$$\gamma_{jk}^{(O)}(c) = \begin{cases} \mathbb{1}\{|\hat{x}_{i_k}^{(b)}(t) - y_{jk}(cT_j^{(O)})| > \epsilon\} & \text{if } s^* > \underline{c}_{jk}^{(O)} T_j^{(O)} \\ \mathbb{1}\{|y_{jk}(\underline{c}_{jk}^{(O)} T_j^{(O)}) - y_{jk}(cT_j^{(O)})| > \epsilon\} & \text{else} \end{cases}$$

where

$$\begin{aligned} t &\triangleq \max_{\underline{c}_{jk}^{(O)} T_j^{(O)} < s + n\Delta t^{(b)} \leq cT_j^{(O)}} \{s + n\Delta t^{(b)} : \beta(s, Z^{(O)}(s)) = 1\} \\ s^* &\triangleq c' T_{j'}^{(O)} \end{aligned} \tag{2.63}$$

where $j' \in \{1, \cdots, M\}/\{j\}$ and $c' \in \mathbb{N}$ are solutions to the equation

$$c' T_{j'}^{(O)} = (\text{argmax } t) - U_{j',c'}^{(O)} \Delta t^{(s)} \tag{2.64}$$

Here, $t$ is the time of the latest broadcast received by transmission time $cT_j^{(O)}$, and $\epsilon > 0$ is some chosen threshold value. We refer to (2.79) as the *smart broadcast feedback triggering strategy.*

Finally, we specifically choose the broadcast rule for a feedback architecture to be as follows:

$$\beta(t, Z^{(O)}(t)) = \begin{cases} \begin{cases} 1 & \text{w.p. } p^{(b)} \\ 0 & \text{w.p. } 1 - p^{(b)} \end{cases} & \text{if } \exists j \in \{1, \cdots, M\}, c \in \mathbb{N} \text{ s.t. } t = cT_j^{(O)} + U_{j,c}^{(O)} \Delta t^{(s)}, \mathcal{G}_j^{(O)}(c) \neq \varnothing \\ 0 & \text{otherwise} \end{cases} \tag{2.65}$$

where $p^{(b)} \in [0,1]$ is constant.

We emphasize that these simplifying assumptions hold only for the scope of Section **??** and not for the rest of the paper unless explicitly stated otherwise.

The assumptions introduced in Section **??** allow us to model the communications over time according to a renewal process with three different phases: 1) "up" when any one of the $M$ sensors is transmitting uplink, "down" when the central processor is broadcasting downlink, and "null" when there is no ongoing communication. Let the sequence of $\{cT_j^{(\chi)}\}_{c\in\mathbb{N}}$ form the arrival times of a renewal process modeling the uplink transmissions of sensor $j$ under architecture $\chi$. For this renewal process, define an *inter-transmission period* of sensor $j$ to be the time interval between two consecutive transmissions of sensor $j$. For each sensor $j \in \{1, \cdots, M\}$ and each component $k \in \{1, \cdots, m_j\}$, further define

$$p_{jk}^{(\chi)}(c) \triangleq \mathbb{P}(\gamma_{jk}^{(\chi)}(c) = 1) \tag{2.66}$$

to be the probability that component $k$ is transmitted uplink by sensor $j$ during inter-transmission period $c$ under strategy $\chi \in \{I, O\}$, using the specific (2.78) for $\chi = I$ and the specific (2.79) for $\chi = O$.

**Lemma 3.** Under Assumption 6, nonfeedback triggering strategy (2.78), and periodic transmission assumption in Definition 7, $p_{jk}^{(I)}(c) = q_{i_k,j}^{(I)}(\theta)$ for all $c \in \mathbb{N}$.

*Proof.* This is obvious because the absolute-difference nonfeedback triggering rule is activated only when there is a $\theta$-change in the environment. ∎

Associated with the renewal process is a reward process for the power consumed by sensor $j$ during each inter-transmission period.

**Definition 8** (Inter-Transmission Reward)**.** Define the *reward* accumulated over the $c$th inter-transmission period $[cT_j^{(\chi)}, (c+1)T_j^{(\chi)})$ as a decomposition of three parts:

$$R_{j,c}^{(\chi)} := U_{j,c}^{(\chi)} \cdot P_U + (\tilde{D}_{j,c}^{(\chi)} + D_{j,c}^{(\chi)}) \cdot nP_D \tag{2.67}$$

where $\chi \in \{I, O\}$ and $U_{j,c}^{(\chi)}$ was defined previously. Here, $D_{j,c}^{(\chi)}$ is the number of broadcast packets triggered from sensor $j$'s transmission, and $\tilde{D}_{j,c}^{(\chi)}$ is the number of broadcasts received by other sensors $j' \in \{1, \cdots, M\}/\{j\}$ during the $c$th inter-transmission period. □

**Lemma 4.** For nonfeedback architectures, $D_{j,c}^{(I)} = \tilde{D}_{j,c}^{(I)} = 0$ for all $j$ and $c$. Consequently, $R_{j,c}^{(I)} = U_{j,c}^{(I)} P_U$.

*Proof.* This is obvious because the central processor does not broadcast under a nonfeedback architecture. ∎

If the reward accumulated over each inter-transmission period were i.i.d. over all $c \in \mathbb{N}$, then we can invoke the renewal reward theorem to compute the long-run average cost of power for sensor $j$. However, the $\{R_{j,c}^{(\chi)}\}_{c\in\mathbb{N}}$ for $\chi \in \{I, O\}$ are not independent nor identically-distributed over all $c$, and this is clear from the distribution of $U_{j,c}^{(\chi)}$.

**Lemma 5.** For any $c \in \mathbb{N}$ and any $\chi \in \{I, O\}$:

$$\mathbb{E}[U_{j,c}^{(\chi)}] = \sum_{k=1}^{m_j} p_{jk}^{(\chi)}(c) \tag{2.68}$$

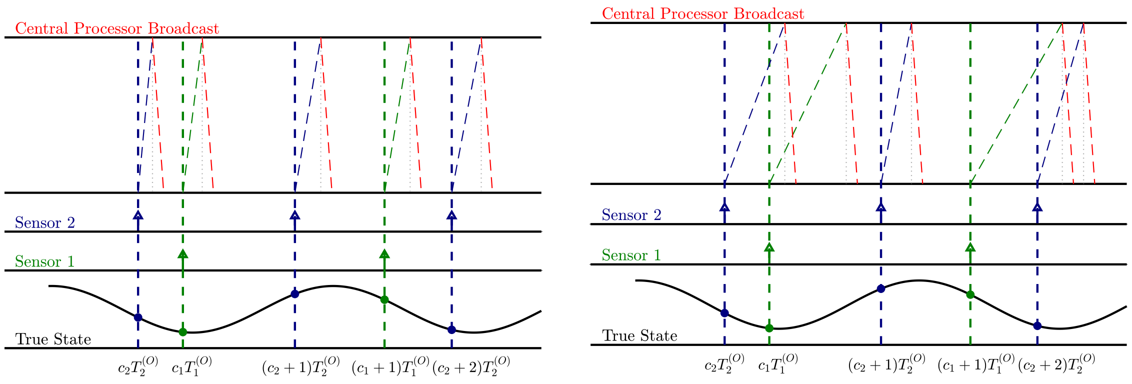

**Figure 2.4:** The time-evolution of the communication between a central processor and $M = 2$ sensors and the case where $p^{(b)}(t) = 1$ for all $t$. [Left] Non-overlapping communications. [Right] General, possibly overlapping communications.

In particular, under the memoryless environment Assumption 6:

$$\mathbb{E}[U_{j,c}^{(I)}] = \sum_{k=1}^{m_j} q_{i_k,j}^{(I)}(\theta) \tag{2.69}$$

*Proof.* Let $X_{jk}^{(\chi)}(c)$ be the indicator random variable which takes value 1 if the $k$th packet of sensor $j$ gets transmitted during its $c$th inter-transmission period. Then (2.68) comes from the fact that $U_{j,c}^{(\chi)} = \sum_{k=1}^{m_j} X_{jk}^{(\chi)}(c)$ and (2.80). Lastly, (2.69) comes from Lemma 3 ■

Given the above setup and assumptions, we are interested in deriving an expression for the long-run average cost of power (uplink + downlink) for each sensor in terms of

It is helpful to adopt a case-by-case analysis, which we divide as follows. First, under the broadcasting strategy (2.65), there are two cases to consider based on the relationship between the transmission/broadcast delays and the baseline transmission periods $T_j^{(O)}$.

- *Case 1: Non-Overlapping Communications.* When the delays are small enough, the central processor's broadcast always arrives in between consecutive transmissions from either sensor. When $M = 2$ sensors and $p^{(b)} = 1$, an illustration of this case is shown in the top subfigure of Figure 2.4.
- *Case 2: General Communications.* On the other hand, when the sensors transmit too rapidly, the central processor's broadcast may not always arrive in a timely manner. In the case of $M = 2$ sensors and $p^{(b)} = 1$, this is illustrated in the bottom subfigure of Figure 2.4: note that at times $c_2T_2^{(O)}$ and $(c_1 + 1)T_1^{(O)}$, the broadcast arrives after transmission occurs.

Second, it also matters whether a component of the central processor's state estimate is influenced by observations received by a single sensor or multiple observations received across multiple sensors.

1. *Case A: Single Sensor.* The state component is sensed by only one of the $M$ total sensors.
2. *Case B: Multiple Sensors.* The state component is sensed by multiple sensors.

We henceforth reference combinations of cases by putting their identifications together, as Case $1A$, $1B$, $2A$, or $2B$. For example, Case $1B$ refers to the case where a state component is sensed by multiple sensors in a way such that the communications among the sensors do not overlap.

**Lemma 6.** For any sensor $j \in \{1, \cdots, M\}$ and $c \in \mathbb{N}$:

$$\mathbb{E}[D_{j,c}^{(O)}] = p^{(b)}\mathbb{P}(U_{j,c}^{(O)} \geq 1) = 1 - \mathbb{P}\left(U_{j,c}^{(O)} = 0\right) = p^{(b)}\left(1 - \prod_{k=1}^{m_j}(1 - p_{jk}^{(O)}(c))\right) \tag{2.70}$$

*Proof.* Due to Assumption 8, any broadcast made in response to sensor $j$'s transmission at time $cT_j^{(O)}$ is made during the $c$th inter-transmission time $[cT_j^{(O)}, (c+1)T_j^{(O)})$. The probability that a transmission is made from sensor $j$ at time $cT_j^{(O)}$ is equivalent to the probability that at least one packet activates the triggering rule. ■

**Setting 1** (Two Sensor Simple Scenario)**.** We consider an architecture with $M = 2$ sensors which have no measurement noise, i.e. $\sigma = 0$ in (2.48). The baseline transmission periods are such that $T_1^{(\chi)} = T_2^{(\chi)} \triangleq T^{(\chi)}$ for $\chi \in \{I, O\}$, where sensor 1 transmits with a shift offset of $(1/2)T^{(\chi)}$ from sensor 2. Note that because $T_1^{(\chi)} = T_2^{(\chi)}$, we have $q_{i_k,1}(\rho) = q_{i_k,2}(\rho) \triangleq q_{i_k}(\rho)$ for all $\rho > 0$.

**Remark 6.** In Setting 4, Case 1 is achieved under the condition

$$m_j \Delta t^{(s)} + n\Delta t^{(b)} \leq \frac{1}{2}T^{(\chi)} \tag{2.71}$$

for both $j \in \{1, 2\}$. □

For Setting 4, we further define the followign two quantities. Let $m' \leq \min(m_1, m_2)$ denote the number of components which are sensed by both sensors 1 and 2, i.e. $m'$ components belong under Case $B$. Also denote

$$\overline{m} \triangleq \min\left\{m \leq \min(m_1, m_2) : m\Delta t^{(s)} + n\Delta t^{(b)} > \frac{1}{2}T^{(\chi)}\right\} \tag{2.72}$$

to be the largest number of packets which can be transmitted without having overlaps (Case 2).

**Assumption 9.** For Setting 4, $\overline{m}$ and $m'$ are such that $\overline{m} \leq \min\{m', m_1 - m', m_2 - m'\}$. □

**Lemma 7.** Under Setting 4, we can simplify (2.68) and (2.70) as follows

$$\mathbb{E}[U_{j,c}^{(O)}] = \sum_{k=1}^{m'} \frac{1}{2} q_{i_k,j}^{(O)}(\epsilon) + \sum_{k=m'+1}^{m_j} q_{i_k,j}^{(O)}(\epsilon) \tag{2.73a}$$

$$\mathbb{E}[D_{j,c}^{(O)}] = p^{(b)}\left[1 - \prod_{k=1}^{m'}\left(1 - \frac{1}{2}q_{i_k,j}^{(O)}(\epsilon)\right)\prod_{k=m'+1}^{m_j}\left(1 - q_{i_k,j}^{(O)}(\epsilon)\right)\right] \tag{2.73b}$$

where $j \in \{1, 2\}$ and $c \in \mathbb{N}$.

*Proof.* In sensor $j \in \{1, 2\}$, there are $m_j - m'$ components which are not sensed by the other sensor $j' \in \{1, 2\}/\{j\}$; hence, their updates are made as if an INformation architecture was being used. By Lemma 3, the transmission probability for components $k \in \{m', \cdots, m_j\}$ components is $q_{i_k,j}^{(O)}(\epsilon)$. On the other hand, for both sensors, there are $m'$ components which are sensed by both sensors; hence, under Setting 4, their updates are made at twice the rate as their updates under an INformation architecture. The transmission probability for components $k \in \{1, \cdots, m'\}$ is then $(1/2)q_{i_k,j}^{(O)}(\epsilon)$. The results then follow from substitution. ■

**Lemma 8.** Under Setting 4 with $q_{i,1} = q_{i,2} \triangleq q$ constant for all $i \in \{1, \cdots, n\}$:

$$\mathbb{P}(U^{(O)}_{j,c} = 0) = \left(1 - \frac{1}{2}q\right)^{m'} (1-q)^{m_j - m'} \tag{2.74a}$$

$$\mathbb{P}(U^{(O)}_{j,c} < \overline{m}) = \sum_{\ell=0}^{\overline{m}-1} \sum_{r=0}^{\ell} \binom{m'}{r} \binom{m_j - m'}{\ell - r} \left(\frac{1}{2}q\right)^{r} \left(1 - \frac{1}{2}q\right)^{m'-r} q^{\ell-r}(1-q)^{m_j-m'-\ell+r} \tag{2.74b}$$

for $j \in \{1, 2\}$ and any $c \in \mathbb{N}$.

*Proof.* Using the argument from the proof of Lemma 7, the probability of transmitting is $q$ for a Case $A$ component and $(1/2)q$ for a Case $B$ component. The results are then derived via a combinatorial argument under Assumption 9. ■

**Theorem 5** (Distribution of $\tilde{D}^{(O)}$ for Setting 4)**.** Consider the Setting 4 with $q_{i,1} = q_{i,2} \triangleq q$ constant for all $i \in \{1, \cdots, n\}$. Then $\mathbb{E}\left[\tilde{D}^{(O)}_{j,c_j}\right]$ takes the same value under Case 2 and Case 1 with Remark 6 satisfied.

$$\mathbb{E}\left[\tilde{D}^{(O)}_{j,c}\right] = p^{(b)} \left(1 - \left(1 - \frac{1}{2}q\right)^{m'} (1-q)^{m_{j'}-m'}\right) \tag{2.75}$$

where $j \in \{1, 2\}$ and $c \in \mathbb{N}$.

*Proof.* Under Case 2, note that $\tilde{D}^{(O)}_{j,c_j} \leq 2$. Furthermore, $\tilde{D}^{(O)}_{j,c}$ is nonzero only if there exists another sensor $j' \in \{1, \cdots, M\}/\{j\}$ and a $c' \in \mathbb{N}$ such that either one of the following two scenarios:

1. it transmits and broadcasts within the $c$th inter-transmission period: $c'T^{(O)}_{j'}, c'T^{(O)}_{j'} + U^{(O)}_{j',c'}\Delta t^{(s)} + n\Delta t^{(b)} \in [cT^{(O)}_{j}, (c+1)T^{(O)}_{j})$ and $U^{(O)}_{j',c'} \geq 1$.
2. it transmits before time $cT^{(O)}_{j}$, but the communication delays allow the broadcast to arrive within the $c$th inter-transmission period: $c'T^{(O)}_{j'} < cT^{(O)}_{j}$ and $c'T^{(O)}_{j'} + U^{(O)}_{j',c'}\Delta t^{(s)} + n\Delta t^{(b)} \in [cT^{(O)}_{j}, (c+1)T^{(O)}_{j})$ and $U^{(O)}_{j',c'} \geq 1$.

This implies that under Case 2:

$$\begin{aligned}\mathbb{E}\left[\tilde{D}^{(O)}_{j,c_j}\right] &= p^{(b)}\mathbb{P}(U_{j',c_{j'}} \geq \overline{m}) \left[(1-p^{(b)})\mathbb{P}(1 \leq U^{(O)}_{j',c_{j'}} < \overline{m}) + \mathbb{P}(U^{(O)}_{j',c_{j'}} = 0) + \mathbb{P}(U^{(O)}_{j',c_{j'}} \geq \overline{m})\right] \\ &+ p^{(b)}\mathbb{P}(1 \leq U^{(O)}_{j',c_{j'}} < \overline{m}) \left[\mathbb{P}(U^{(O)}_{j',c_{j'}} < \overline{m}) + (1-p^{(b)})\mathbb{P}(U^{(O)}_{j',c_{j'}} \geq \overline{m})\right] \\ &+ 2p^{(b)2}\mathbb{P}(1 \leq U^{(O)}_{j',c_{j'}} < \overline{m})\mathbb{P}(U^{(O)}_{j',c_{j'}} \geq \overline{m})\end{aligned} \tag{2.76}$$

Under Case 1, $\tilde{D}^{(O)}_{j,c_j} \leq 1$ and only the first scenario is possible because of (2.71). Simplifying (2.76) using Lemma 8 and the fact that $\mathbb{P}(U^{(O)}_{j,c} \geq \overline{m}) = 1-$ (2.74b) and $\mathbb{P}(1 \leq U^{(O)}_{j,c} < \overline{m}) =$ (2.74b)$-$ (2.74a) yields (2.75). ■

**Remark 7.** Theorem 5 has a very straightforward interpretation: it shows that under symmetric environments and symmetric communication structures between the INformation and OUTformation architectures, Case 1 and Case 2 are indistinguishable. □

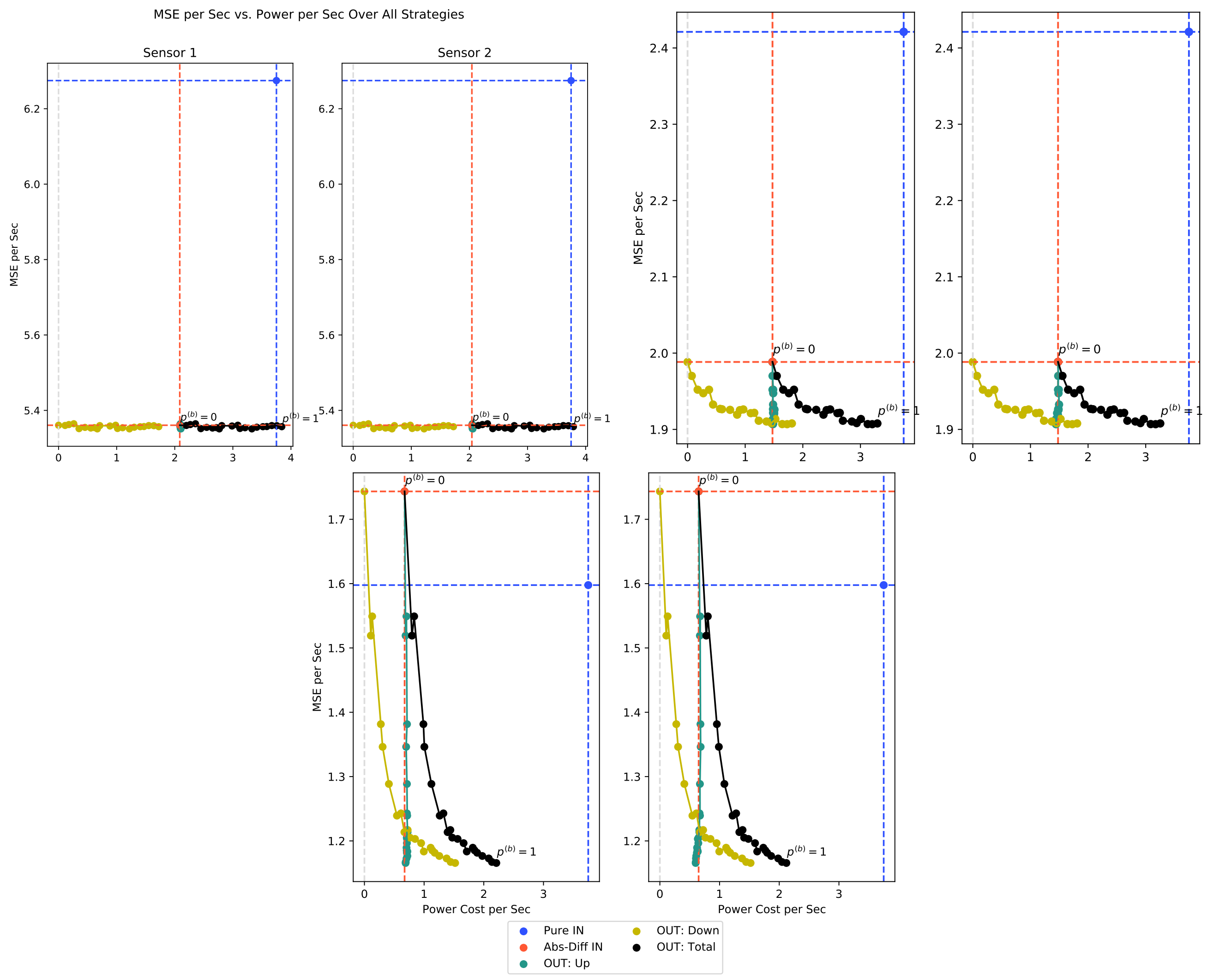

**Figure 2.5:** Plots of the MSE per second vs cost of power per second for both sensors 1 and 2, averaged over 40 Monte-Carlo simulations. For the smart OUTformation architecture, the broadcast probability $p^{(b)}$ is ranged over $\{0, 0.05, 0.1, \cdots, 1\}$, and we make separate curves for the uplink, downlink, and the total (uplink + downlink) costs of power. Each row of subfigures differ by the parameters of the Markov chain environment. [Top Row] $\Delta t = 1, p = 0.9$, the fastest-changing environment. [Middle Row] $\Delta t = 2, p = 0.9$. [Bottom Row] $\Delta t = 3, p = 0.7$, the slowest-changing environment.

**Setting 2** (No-Noise, Periodic Transmissions)**.** Neither of the sensors are corrupted with white noise, i.e. $\sigma = 0$. The environment evolves as the Markov chain, random-walk-like environment from Setting 3 with stepsize $C(c) = 1$ fixed constant for all $c \in \mathbb{N}$ and $q_i = p/2n$ for each $i \in \{1, \cdots, n\}$. At every $T_j^{(\chi)}$ seconds, sensor $j$ checks the validity of the triggering rule under architecture $\chi \in \{I, O\}$, and makes a transmission to the central processor; in this section, we refer to $T_j^{(\chi)}$ as the *baseline transmission period* of sensor $j$. We choose triggering strategy threshold values $\theta = \epsilon = 0.9$.

**Simulation 1 Parameters**: $\Delta t = 2, p = 0.9$ (repeated from previous prose). $T_1^{(I)} = T_1^{(O)} = 20, T_2^{(I)} = T_2^{(O)} = 20$ with shift offset 10. $\Delta t^{(s)} = 2, \Delta t^{(b)} = 0.1, P_U = 5, P_D = 1$.

The results of the experiments over multiple different Markov chain environments are shown in Figure 2.5 and Figure 2.7. As expected, $OUT(\epsilon)$ with $p^{(b)} = 0$ coincides with the point for $IN(\theta)$ because no broadcasts are being made. For the values chosen to generate Figure 2.5, $OUT(\epsilon)$ never beats $IN(\theta)$ in

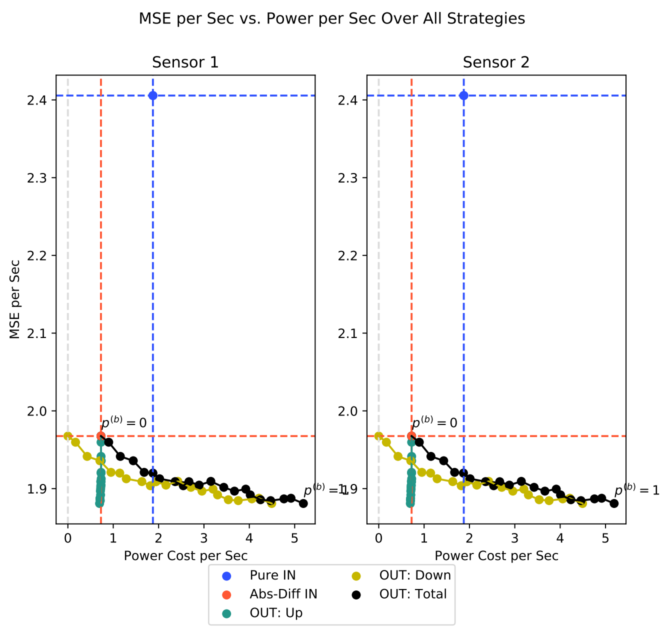

**Figure 2.6:** Same experiment performed in the second row of Figure 2.5 ($\Delta t = 2, p = 0.9$) except with $P_U \leq P_D$.

terms of the power cost while $IN(\theta)$ never beats $OUT(\epsilon)$ in terms of the MSE. This implies that for extremely fast environments like $\Delta t = 0.1, p = 0.9$, the communication delays of the system prevent the broadcast from being useful information. In the case of the stupid OUTformation strategy $OUT_0(\epsilon)$, we note that the total power cost traces out a parabolic curve shape rather than a logarithmic. See the top figures of Figure 2.7. This is because as we decrease the downlink power cost, the broadcast compared against the current observation gets more outdated, which causes the sensors to transmit more and increase the uplink power cost.

Note that these relationships between $OUT(\epsilon)$ and $IN(\theta)$ may change with different values of $T_1^{(\chi)}$, $T_2^{(\chi)}$, $\Delta t^{(s)}$, $\Delta t^{(b)}$, $P_U$, $P_D$, which depend on the hardware specifications of the sensors and the central processor. In fact:

- When changing the per-packet communication delay values such that $\Delta t^{(s)} < \Delta t^{(b)}$ while keeping all other parameters the same, the tradeoff space in the second row of Figure 2.5 resembles exactly the tradeoff space in the first row of Figure 2.5. This is expected since $OUT(\epsilon)$ behaves exactly like $IN(\theta)$ when the broadcast is more outdated than the previous transmission, which occurs more often with $\Delta t^{(s)} < \Delta t^{(b)}$. Hence, even though $OUT(\epsilon)$ spends more total power than $IN(\theta)$, the MSE of $OUT(\epsilon)$ never exceeds that of $IN(\theta)$. We even observe the same trend comparing the top and bottom rows of Figure 2.7 for $OUT_0(\epsilon)$.
- When changing the per-packet cost of power such that $P_U \leq P_D$, it is even possible for $OUT(\epsilon)$ to exceed $IN_0$ in total power cost; see Figure 2.6. This is expected because the central processor naively broadcasts the entire $n$-dimensional vector down to the sensors, even the components which are irrelevant to the sensor. With an appropriate choice of $P_D \geq P_U$, we will certainly have $nP_D \geq m_j P_U$ for $j \in \{1, 2\}$.

For $OUT(\epsilon)$ applied to slower-changing environments, a better balance between the MSE and the cost of power may be obtained by choosing $p^{(b)}$ more appropriately.

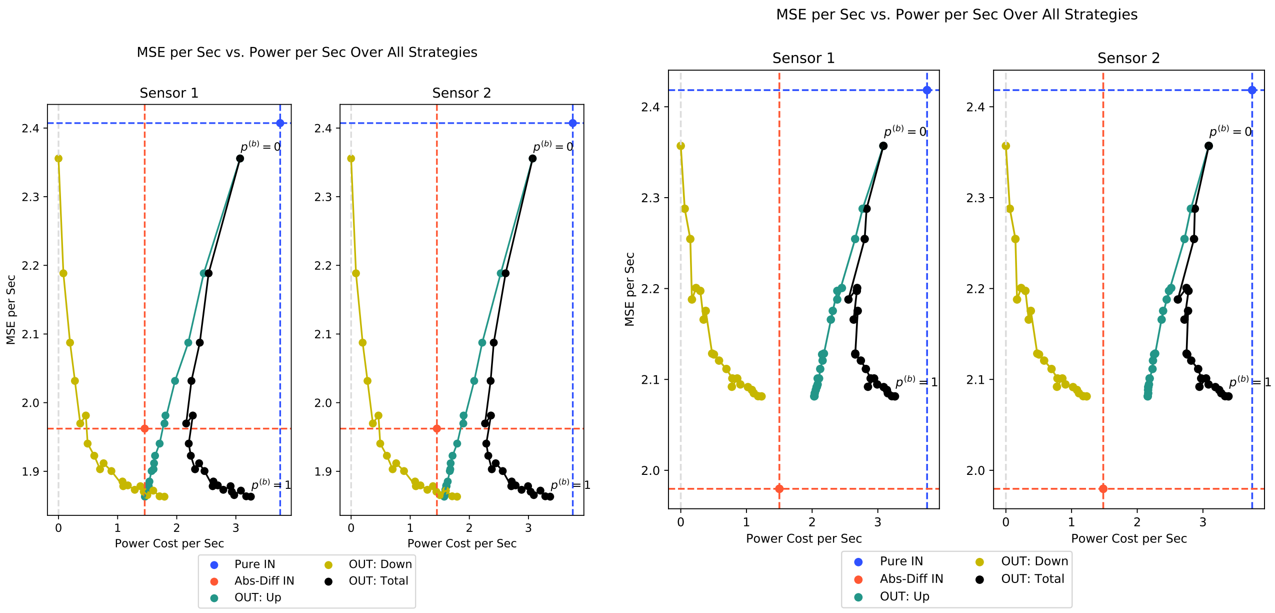

**Figure 2.7:** [Top] A plot of the MSE per second vs cost of power per second for both sensors 1 and 2, averaged over 40 Monte-Carlo simulations for $OUT_0(\epsilon)$ with $\Delta t = 2, p = 0.9$. [Bottom] Same type of figures in the top row with the same parameters used except $\Delta t^{(s)} = 2, \Delta t^{(b)} = 3$, i.e., $\Delta t^{(s)} < \Delta t^{(b)}$.

On environment $\Delta t = 5, p = 0.4$, there is a significant difference in trade-off space when we take $\hat{\mathbf{x}}_0 = \mathbf{x}_0$ versus randomly initializing $\hat{\mathbf{x}}_0 \neq \mathbf{x}_0$. See Figure 2.8. The complete change in positioning between the pure IN point and the abs-diff IN point is particularly concerning, and may be the result of a bug. However, faster-changing environment $\Delta t = 2, p = 0.9$ does not suffer the same problem: tradeoff spaces are consistent regardless of $\hat{\mathbf{x}}_0 = \mathbf{x}_0$ or $\hat{\mathbf{x}}_0 \neq \mathbf{x}_0$. Conjecture: there are less transitions with $\Delta t = 5, p = 0.4$, and so even having that extra transition at the beginning adds a significant amount to an MSE/power cost which is mostly zero over time. This is suggested by the absolute values of MSE being less for $\Delta t = 5, p = 0.4$ than for $\Delta t = 2, p = 0.9$. This conjecture is correct. If we compare $\hat{\mathbf{x}}_0 = \mathbf{x}_0$ case with $\hat{\mathbf{x}}_0 \neq \mathbf{x}_0$ for the environment $\Delta t = 2, p = 0.7$, which is slower than $\Delta t = 2, p = 0.9$ but significantly faster than $\Delta t = 5, p = 0.4$, we obtain no difference in the relative positions of the $IN_0$ point and the $IN(\theta)$ point. Hence, in slow-changing environments, the MSE is mostly zero over time, so even starting on the wrong initial estimate yields a significant change in the tradeoff space. This reasoning is further supported by looking at $\Delta t = 5.0, p = 1.0$ and increasing the simulation time from $T_{\text{sim}} = 200$ to $T_{\text{sim}} = 1000$: the case $T_{\text{sim}} = 200$ has $IN(\theta)$ point placed higher than $IN_0$ point, but the case $T_{\text{sim}} = 1000$ has $IN(\theta)$ perform better in both aspects compared to $IN_0$.

We emphasize that OUTformation offers an advantage for each sensor's estimate of the central processor's estimate. Under any INformation strategy, a sensor's belief about the central processor's estimate about state component $i_k$ will be its own local transmission of state component $i_k$, i.e., sensor $j$ knows it sent the central processor $y_{jk}$ about $i_k$, so it believes the central processor knows $y_{jk}$. However, under any OUTformation strategy, a sensor will understand that other sensors may have provided more up-to-date information about $i_k$ to the central processor. The advantage of the broadcast, assuming it arrives quickly enough under reasonably small power cost, is that sensor $j$ may know that the central processor already knows $y_{j'k}$ from another sensor $j'$, which is potentially more updated than the previous transmission $y_{jk}$.

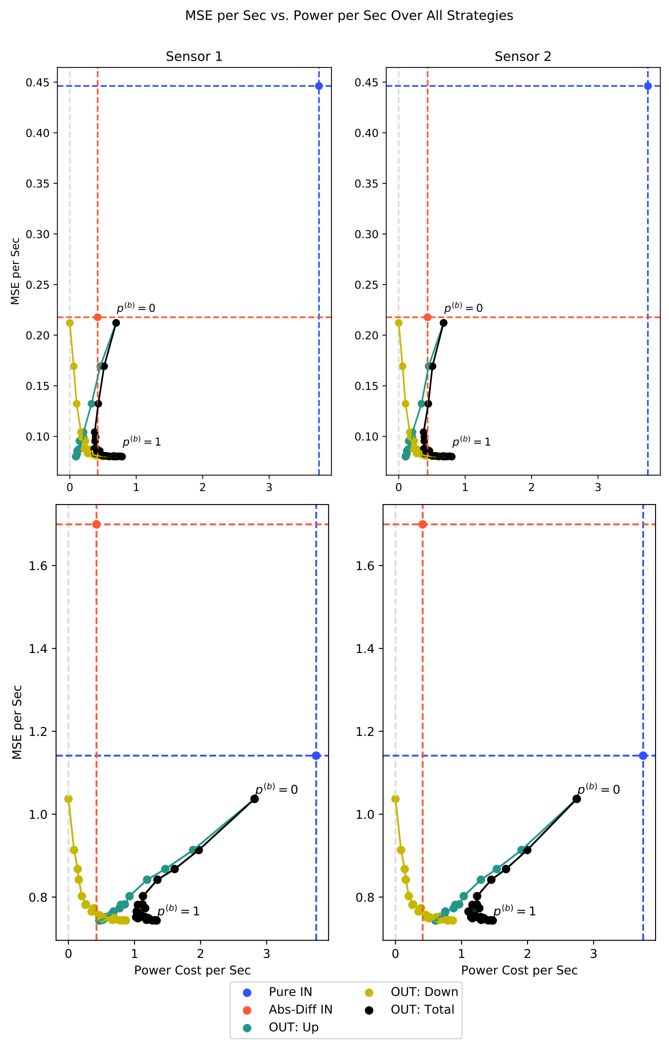

**Figure 2.8:** Same type of figures as the top row of Figure 2.7 with the same parameters used, except $\Delta t = 5, p = 0.4$. [Top Row] $\hat{\mathbf{x}}_0 = \mathbf{x}_0$. [Bottom Row] $\hat{\mathbf{x}}_0 \neq \mathbf{x}_0$.

### 2.4.3 Event-Triggered Communication

Under further simplifying assumptions on the model of Section **??** and on the environment it is surveying, we can derive some theoretical guarantees on the performance of distributed state estimation using one type of OUTformation architecture versus one type of INformation architecture. In this section, we provide an analysis on tradeoff guarantees for the specific setting described below.

**Setting 3** (Periodic- Bernoulli Environment)**.** The true environment evolves as a Markov Chain, random-walk-like environment with a high-dimensional vector state of length $n \in \mathbb{N}$ and transitions distributed according to

$$\mathbf{x}(c\Delta t) = \begin{cases} \mathbf{x}((c-1)\Delta t) & \text{w.p. } 1 - \sum_{i=1}^{n} q_i \\ \mathbf{x}(t) \pm (C(c), 0, \cdots, 0)^T & \text{w.p. } \frac{q_1}{2} \\ & \vdots \\ \mathbf{x}(t) \pm (0, \cdots, 0, C(c))^T & \text{w.p. } \frac{q_n}{2} \end{cases} \tag{2.77}$$

where $c \in \mathbb{N}$, $\Delta t \in \mathbb{R}^+$ is the period of the environment, and $C(c) \in [\rho, \infty)$ is a positive random variable which always takes value larger than some fixed constant $\rho > 0$. Essentially, once every $\Delta t$ seconds, the environment changes by a step size of $C(c)$ in randomly-chosen component $i \in \{1, \cdots, n\}$ with probability $q_i$, and remains at its current position otherwise. We henceforth refer to this type of environment as the *periodic-Bernoulli environment* with period $\Delta t$ and probability profile $[q_1, \cdots, q_n]$.

Under Setting 3, we can also ensure that potential collisions with other sensors never occur by having each sensor $j$ choose backoff times $\{B_{jk}(t)\}_{k=1}^{m_j}$ depending on whether or not each component $k$ is observed by other sensors.

**Definition 9** (Shared vs. Unshared Components)**.** A component $i \in \{1, \cdots, n\}$ in the full vector state $\mathbf{x}(t) \in \mathbb{R}^n$ is is said to be *shared* by two sensors $j, j' \in \{1, \cdots, M\}$ if both sensors $j$ and $j'$ are observing it according to their respective measurement equations (2.48). Components can be shared by more than two sensors as well. In contrast, an *unshared* component is an $i \in \{1, \cdots, n\}$ which is observed by only a single sensor. □

**Assumption 10** (Sensors Know Shared Components)**.** At the start of the simulation, we assume the sensors know which of their components are shared and which are unshared. This applies to both INformation and OUTformation architectures. □

For each sensor $j$, we define the set $\mathcal{U}_j \subseteq \{1, \cdots, m_j\}$ to be the set of sensor $j$'s unshared components, and $\mathcal{S}_j \triangleq \mathcal{U}_j^c$ to be the set of components sensor $j$ shares with other sensors. We then assign random backoff time $B_{jk}(t) = B_{\mathcal{U}_j}(t)$ to each unshared component $k \in \mathcal{U}$, and $B_{jk}(t) = B_{\mathcal{S}_j}(t)$ to each $k \in \mathcal{S}_j$. Because there is only one component change of the true state per $\Delta t$, there are no collisions among transmissions of unshared components, and so we can set $B_{\mathcal{U}_j}(t) = 0$ for all $j$ and $t > 0$.

For both the OUTformation and INformation architectures considered in this section, we specify the triggering rule from Figure 2.3 in the following ways. We specifically choose the triggering rule for an INformation architecture to be the following at time $t > 0$:

$$\gamma_{jk}^{(I)}(t, B_{jk}(t)) = \mathbb{1}\left\{|y_{jk}(s) - y_{jk}(t)| \geq \theta\right\} \tag{2.78}$$

Here, $B_{jk}(t)$ is the random backoff time generated at time $t$, $s < t + B_{jk}(t)$ is the time of the previous transmission of component $k \in \{1, \cdots, m_j\}$ by sensor $j$, and the threshold parameter $\theta > 0$ is chosen by design. Essentially, for each component $k$, (2.78) compares the current observation at time $t$ to the previous transmission of packet $k$ at time $s$; a transmission is made if the difference is deemed "large enough" ($> \theta$). We note that as a result, the random backoff time does not affect the triggering criterion because the sensor does not collect any new information during the backoff time. We henceforth refer to (2.78) as the *absolute-difference INformation triggering* strategy.

For OUTformation architectures, we consider a similar thresholding criterion, but we also use the broadcast $\hat{\mathbf{x}}^{(b)}(t)$ of the central processor's estimate. At time $t > 0$, the triggering rule for OUTformation architectures is given by

$$\gamma_{jk}^{(O)}(t, B_{jk}(t)) = \begin{cases} \mathbb{1}\{|\hat{x}_{i_k}^{(b)}(t + B_{jk}(t)) - y_{jk}(t)| \geq \epsilon\} & \text{if } s^* \geq s \wedge k \in \mathcal{S}_j \\ \gamma_{jk}^{(I)}(t, B_{jk}(t)) & \text{if } s^* < s \vee k \in \mathcal{U}_j \end{cases} \tag{2.79}$$

Here, $s < t$ is the time of the previous transmission of component $k \in \{1, \cdots, m_j\}$ by sensor $j$, $s^* < t$ is the time of the transmission of component $k \in \{1, \cdots, m_{j'}\}$ from any sensor $j' \in \{1, \cdots, M\}/\{j\}$ which created the broadcast value $\hat{x}_{i_k}^{(b)}(t)$, and $\epsilon > 0$ is some chosen threshold value. Essentially, if the latest broadcast contains data that is more outdated than the sensor's own previous transmission, we revert back to the absolute-difference INformation triggering strategy (2.78). We further note that (2.79) is simply equivalent to (2.78) if component $i_k \in \{1, \cdots, n\}$ is sensed by only one sensor. We refer to (2.79) as the *smart broadcast OUTformation triggering strategy.*

For each sensor $j \in \{1, \cdots, M\}$ and each component $k \in \{1, \cdots, m_j\}$, further define

$$p_{jk}^{(\chi)}(t, B_{jk}(t)) \triangleq \mathbb{P}(\gamma_{jk}^{(\chi)}(t, B_{jk}(t)) = 1) \tag{2.80}$$

to be the probability that component $k$ is transmitted uplink by sensor $j$ at time $t$ under architecture $\chi$, using the specific (2.78) for $\chi = I$ and the specific (2.79) for $\chi = O$.

Finally, we designate the broadcast rule for an OUTformation architecture to be as follows. Let $\mathcal{C}(t)$ denote the set of packets the central processor receives at time $t$. Because there is no benefit to sensors receiving updates for components they do not observe, a broadcast is not made for any packets corresponding to unshared components. Updates in $\mathcal{C}(t)$ for shared components are automatically broadcast to all the sensors.

In our presentation of the theoretical results, we adhere to the following architecture setup.

**Setting 4** (Two Sensor Architectures)**.** We consider architectures of Figure 2.3 with $M = 2$ sensors sensing the environment of Setting 3. Sensor $j \in \{1, 2\}$ has $m_j \in \mathbb{N}$ components such that $m' < \min(m_1, m_2)$ are shared; for notation simplicity, the observation vector $\mathbf{y}_j(t)$ is formatted such that the first $m'$ components are shared for both $j \in \{1, 2\}$. For shared components $k \in \{1, \cdots, m'\}$, sensor 1 transmits immediately (at the same time as its unshared components), while sensor 2 waits an additional backoff time of $B(t) \sim p_B(t, \cdot)$ where $p_B(t, s) \triangleq \mathbb{P}(B(t) \leq s)$ is the probability distribution of backoff time $B(t)$ to transmit $k$. Both sensors have the same white noise covariance $\Sigma \triangleq \Sigma_1 = \Sigma_2$, same sampling rate $T^{(s)} \triangleq T_1^{(s)} = T_2^{(s)}$, and the same amount of memory in their observation tables $H^{(s)} \triangleq H_1^{(s)} = H_2^{(s)}$. We choose $T^{(s)} > 0$ such that $\Delta t$ from Setting 3 is divisible by $T^{(s)}$.

**Assumption 11** (Small Horizon Observation Table)**.** The value $H^{(s)}$ is chosen conveniently such that $(H^{(s)} - 1)T^{(s)} \leq \Delta t$; this is so that only at most one change in each component of the true environment state is detected at each update of each sensor's observation table. □

**Assumption 12** (Parameter Values)**.** The threshold parameters for both triggering strategies (2.78) and (2.79) are chosen such that $\theta = \epsilon = \rho$, where $\rho$ is from Setting 3. □

**Assumption 13** (Capped Backoff Time)**.** The distribution of the backoff time $p_B(t, s)$ for sensor 2 is such that $p_B(t, \Delta t - \Delta t^{(s)} - \Delta t^{(b)}) = 1$ for all $t > 0$. □

Under the above environment and architecture setup, we can now derive analytical expressions pertaining to the long-run average cost of total power (uplink + downlink) for each sensor in terms of the specified parameters. We model each component's communications over time of each individual component $i \in \{1, \cdots, n\}$ of the full environment state according to a renewal process with three different phases: 1) "up" when a sensor is transmitting uplink, "down" when the central processor is broadcasting downlink, and "null" when there is no ongoing communication. Let the sequence of $\{T_j^{(\chi)}(c)\}_{c\in\mathbb{N}}$ form the renewal times of a renewal process modeling the uplink transmissions of sensor $j \in \{1, 2\}$ under architecture $\chi \in \{I, O\}$ during the interval of time $[c\Delta t, (c + 1)\Delta t)$; we henceforth refer to $[c\Delta t, (c + 1)\Delta t)$ as the *environment interval* $c$.

With the renewal times $\{T_j^{(\chi)}(c)\}_{c\in\mathbb{N}}$, we associate a reward process $\{R_j^{(\chi)}(c)\}_{c\in\mathbb{N}}$ for the total uplink and downlink power consumed by sensor $j$ under architecture $\chi$ during environment interval $c$. For each $j, \chi, c$, we define $U_j^{(\chi)}(c) \in \mathbb{N}$ to be the number of uplink packets transmitted, $D_j^{(\chi)}(c) \in \mathbb{N}$ to be the number of downlink packets received by sensor $j$ under architecture $\chi$ as a result of some transmission made during environment interval $c$. Under Setting 3 and Setting 4, we have $U_j^{(\chi)}(c) \leq (\Delta t/T^{(s)}) - 1$ for all $c \in \mathbb{N}$. Clearly, for INformation architectures, $D_j^{(I)}(c) = \tilde{D}_j^{(I)}(c) = 0$ for all time $c \in \mathbb{N}$ and all $j \in \{1, 2\}$.

The total amount of power consumed by sensor $j$ under architecture $\chi$ over environment interval $c \in \mathbb{N}$ can now be represented as a decomposition of two parts:

$$R_j^{(\chi)}(c) = P_U U_j^{(\chi)}(c) + P_D D_j^{(\chi)}(c) \tag{2.81}$$

We can further provide statistics about the total power consumption by considering packets generated from components which are shared versus unshared.

**Component $i$ is unshared** First, we consider the amount of power consumed by sensor $j \in \{1, 2\}$ throughout communication of unshared packets $k \in \{m' + 1, \cdots, m_j\}$. The following result holds also for general number of sensors $M \geq 2$.

**Theorem 6** (Power from Unshared Components)**.** Under the assumptions in Section **??**, the total amount of power spent on the unshared components of any sensor $j$ is the same for both INformation and OUTformation architectures. Moreover, the expected total power is given by

$$\begin{aligned}&\mathbb{E}[R_{j,m'+1:m_j}^{(\chi)}(c)]\\ &\quad = P_U \sum_{r=0}^{(\Delta t/T^{(s)})-1} \sum_{k=m'+1}^{m_j} p_{jk}^{(I)}(c\Delta t + rT^{(s)}, 0)\end{aligned} \tag{2.82}$$

where $R_{j,m'+1:m_j}^{(\chi)}(c)$ defines the part of the power (2.81) contributed by components $m' + 1$ to $m_j$, $i_k \in \{1, \cdots, n\}$ corresponds to the position of component $k$ in the full vector state, and the other parameters come from Setting 3 and Setting 4.

*Proof.* Under the broadcasting rule from Section **??**, the triggering criteria for unshared components defaults to the absolute-difference criteria (2.78) under both architectures $\chi \in \{I, O\}$. Moreover, there is zero backoff time for both $j \in \{1, 2\}$ and unshared $k$ from Setting 4. To show (2.82), let $X_{jk}^{(\chi)}(c)$ be the indicator random variable which takes value 1 if the $k$th packet of sensor $j$ gets transmitted during environment interval $c$. Note that $X_{jk}^{(\chi)}(t) = 1$ if component $k$ is transmitted by sensor $j$. The result finally follows from (2.80) and observing that $U_{j,m'+1:m_j}^{(\chi)}(c) = \sum_{k=m'+1}^{m_j} X_{jk}^{(\chi)}(c)$, where $U_{j,m'+1:m_j}^{(\chi)}(c)$ denotes the number of uplink transmissions of components $m'+1$ to $m_j$ during environment interval $c \in \mathbb{N}$. ■

**Component $i$ is sensed by both sensors** As mentioned in Setting 4, for components $k \in \{1, \cdots, m'\}$ which are sensed by both sensors, sensor 1 always transmits immediately, while sensor 2 always performs random backoff with duration $B(t) \sim p_B(t, \cdot)$. Define $\tilde{y}_{jk}^{(\chi)}(c)$ to be the most up-to-date previous transmission or latest broadcast value of component $k$ of sensor $j$ obtained before time $c\Delta t$ under architecture $\chi \in \{I, O\}$. Essentially, $\tilde{y}_{jk}^{(\chi)}(c)$ is the value sensor $j$ compares against its current observation $y_{jk}(t), t \in [c\Delta t, (c+1)\Delta t)$ in the triggering rule for architecture $\chi$ if no communications have occurred in the subinterval of time $[c\Delta t, t]$.

The following cases are derived for only shared components between both sensors 1 and 2 under the OUTformation architecture; the INformation architecture and the the case of unshared components are much simpler.

1. *No uplink transmissions [Sensor 1].*
   (a) *sensor* 2 *has no transmissions*:
      - *with good noise*: no true change (prob. $(1-q_{i_k})$) + not triggered (for all $r \in \{0, \cdots, \Delta t/T^{(s)} - 1\}$, $|\tilde{y}_{1k}(c) - y_{1k}(c\Delta t + rT^{(s)})| < \epsilon$).
      - *with bad noise*: true change (prob. $q_{i_k}$) + not triggered (see above).

   (b) *sensor* 2 *backs off and transmits* ***once*** *with good/bad noise*: suppose sensor 2 transmits $y_{2k}(c\Delta t + r_2 T^{(s)})$; then the corresponding broadcast arrives at time $s_2 \triangleq c\Delta t + r_2 T^{(s)} + B(c\Delta t + r_2 T^{(s)}) + \Delta t^{(s)} + \Delta t^{(b)}$. Denote $\underline{s}_1 < s_2$ to be the last potential transmission time of sensor 1 using $\tilde{y}_{1k}(c)$, and denote $\overline{s}_1 \geq s_2$ to be the first potential transmission time of sensor 1 using $y_{2k}(c\Delta t + r_2 T^{(s)})$. Note that we can represent $\underline{s}_1 \triangleq c\Delta t + \underline{r}_1 T^{(s)}$, likewise $\overline{s}_1$ with $\overline{r}_1$, and that $\overline{r}_1 = \underline{r}_1 + 1$ always. Then we need no outdated triggering + no updated triggering (for all $r \leq \underline{r}_1$, $|\tilde{y}_{1k}(c) - y_{1k}(c\Delta t + rT^{(s)})| < \epsilon$ AND for all $r \geq \overline{r}_1$, $|y_{2k}(c\Delta t + r_2 T^{(s)}) - y_{1k}(c\Delta t + rT^{(s)})| < \epsilon$).

   (c) *sensor* 2 *transmits* ***more than once*** *with good/bad noise.* Analysis is extendable from above Case 1*b*) where sensor 2 transmits once with good/bad noise.

2. *No uplink transmissions [Sensor 2].*
   (a) *sensor* 1 *has no transmissions + Case* 1) *for sensor* 2 *with "not triggered" meaning "not backed off".*

   (b) *sensor* 1 *transmits* ***once*** *with good/bad noise*: suppose sensor 1 transmits $y_{1k}(c\Delta t + r_1 T^{(s)})$ once; then the corresponding broadcast arrives at time $s_1 \triangleq c\Delta t + rT^{(s)} + \Delta t^{(s)} + \Delta t^{(b)}$. Denote $\underline{s}_2 < s_1$ to be the maximum time of sensor 2's backoff transmission creation time using $\tilde{y}_{2k}(c)$, and denote $\overline{s}_2 \geq s_1$ to the first time of sensor 2's backoff transmission creation time using $y_{1k}(c\Delta t + r_1 T^{(s)})$. Note that we can represent $\underline{s}_2 \triangleq c\Delta t + \underline{r}_2 T^{(s)}$, likewise $\overline{s}_2$ with $\overline{r}_2$, and that $\overline{r}_2 = \underline{r}_2 + 1$ always.
      - no backoff trigger (for all $r \leq \underline{r}_2$, $|\tilde{y}_{2k}(c) - y_{2k}(c\Delta t + rT^{(s)})| < \epsilon$ AND for all $r \geq \overline{r}_2$, $|y_{1k}(c\Delta t + r_1 T^{(s)}) - y_{2k}(c\Delta t + rT^{(s)})| < \epsilon$)
      - backoff trigger + updated broadcast terminates it (there exists $r \leq r_1$ such that $|\tilde{y}_{2k}(c) - y_{2k}(c\Delta t + rT^{(s)})| < \epsilon$ AND $B(c\Delta t + rT^{(s)}) > (r_1 - r)T^{(s)} + \Delta t^{(s)} + \Delta t^{(b)}$ AND $|y_{1k}(c\Delta t + r_1 T^{(s)}) - y_{2k}(c\Delta t + rT^{(s)})| < \epsilon$)

   (c) *sensor* 1 *transmits* ***more than once*** *with good/bad noise.* Analysis is extendable from above case sensor 1 transmits once with good/bad noise.

3. Unfinished. Easier to see with pictures. *Both sensors transmit **once***. Note that a transmission can occur at any time $c\Delta t + rT^{(s)}, r \in \{0, \cdots, \Delta t/T^{(s)} - 1\}$ during environment interval $c$ because of noise. Of course, by Setting 3, the best MSE occurs when $r = 0$. Suppose sensor $j$ transmits a component which was observed at time $c\Delta t + r_j T^{(s)}$, for each $j \in \{1, 2\}$. Then the broadcast for sensor 1 occurs at time $s_1 \triangleq c\Delta t + r_1 T^{(s)} + \Delta t^{(s)} + \Delta t^{(b)}$ and the broadcast for sensor 2 occurs at time $s_2 \triangleq c\Delta t + r_2 T^{(s)} + B(c\Delta t + r_2 T^{(s)}) + \Delta t^{(s)} + \Delta t^{(b)}$ (both were defined in previous cases).

   (a) *same component* $r_1 = r_2 \triangleq r'$. Then the broadcast for sensor 1 always arrives earlier than the broadcast for sensor 2. First, we need no triggering before and after $r'$ (for all $r < r'$, $|\tilde{y}^{(\chi)}_{jk}(c) - y_{jk}(c\Delta t + rT^{(s)})| < \epsilon$ AND for all $r \in \mathbb{N}$ such that $s_1 \leq c\Delta t + rT^{(s)} < s_2$, $|y_{1k}(c\Delta t + r'T^{(s)}) - y_{jk}(c\Delta t + rT^{(s)})| < \epsilon$ AND all $r \in \mathbb{N}$ such that $c\Delta t + rT^{(s)} \geq s_2$, $|y_{2k}(c\Delta t + r'T^{(s)}) - y_{jk}(c\Delta t + rT^{(s)})| < \epsilon$ ). AND we also need triggering to occur precisely for the observations generated at $r'$:

      - *sensor 1's round-trip is longer than sensor 2's backoff duration.* ($B(c\Delta t + r'T^{(s)}) < s_1 - c\Delta t - r'T^{(s)}$ AND $|\tilde{y}^{(\chi)}_{1k}(c) - y_{1k}(c\Delta t + r'T^{(s)})| \geq \epsilon$ AND $|\tilde{y}^{(\chi)}_{2k}(c) - y_{2k}(c\Delta t + r'T^{(s)})| \geq \epsilon$).
      - *sensor 2's backoff duration is longer than sensor 1's round-trip.* ($B(c\Delta t + r'T^{(s)}) \geq s_1 - c\Delta t - r'T^{(s)}$ AND $|\tilde{y}^{(\chi)}_{1k}(c) - y_{1k}(c\Delta t + r'T^{(s)})| \geq \epsilon$ AND $|y_{1k}(c\Delta t + r'T^{(s)}) - y_{2k}(c\Delta t + r'T^{(s)})| \geq \epsilon$).

   (b) *sensor 1 earlier observation* $r_1 < r_2$. Again, the broadcast for sensor 1 always arrives earlier than the broadcast for sensor 2. First, we need no triggering before $r < r_1$, in between $r_1 < r < r_2$, and after $r > r_2$ (for all $r < r_1$, $|\tilde{y}^{(\chi)}_{jk}(c) - y_{jk}(c\Delta t + rT^{(s)})| < \epsilon$ AND for all $r \in \mathbb{N}$ such that $s_1 \leq c\Delta t + rT^{(s)} < s_2$, $|y_{1k}(c\Delta t + r'T^{(s)}) - y_{jk}(c\Delta t + rT^{(s)})| < \epsilon$ AND all $r \in \mathbb{N}$ such that $c\Delta t + rT^{(s)} \geq s_2$, $|y_{2k}(c\Delta t + r'T^{(s)}) - y_{jk}(c\Delta t + rT^{(s)})| < \epsilon$ ). AND we also need triggering to occur precisely for the observations generated at $r'$:

      - $s_1 > s_2 - \Delta t^{(s)} - \Delta t^{(b)}$.
      - $s_1 \leq s_2 - \Delta t^{(s)} - \Delta t^{(b)}$.

   (c) *sensor 2 earlier observation* $r_2 < r_1$.

      - $s_1 > s_1 - \Delta t^{(s)} - \Delta t^{(b)}$.
      - $s_1 \leq s_1 - \Delta t^{(s)} - \Delta t^{(b)}$.

4. *Both sensors transmit **more than once***. Let $1 \leq N_{jk} \leq \Delta t/T^{(s)} - 1$ be the number of transmissions, made at times $\{c\Delta t + rT^{(s)}\}$ for $N_{jk}$ values of $r \in \{0, \cdots, \Delta t/T^{(s)} - 1\}$. Due to Setting 3, only at most one transmission is a result of good noise; all other transmissions are bad.

   - *all bad noise*: no true change + $N_{jk}$ triggered.
   - *one good noise*: true change + earliest one of the $N_{jk}$ triggered. All later triggers are bad.

   Note that for both abs-diff IN and OUT the triggering rule changes sequentially with each of the $N_{jk}$ transmissions made in environment interval $c$. Analysis is extendable from a combination of the previous considerations.

Emphasize that the central processor knows 1) only one component changes every $\Delta t$, and 2) which components are shared among sensors. It uses that data to coordinate the best transmission strategy among sensors (i.e. zero backoff for unshared, positive backoff for shared). This transmission strategy is told to the sensors at the beginning of the simulation, but nothing else, because the lightweight sensors have the capacity for nothing else. Transmission strategy is designated by the central processor.

For each component $k \in \{1, \cdots, m'\}$, we have the following casewise analysis of the relationship between sensor 2's backoff duration $B$, and the duration of one round-trip of communications from sensor 1.

- *Sensor* 1 *does not transmit component $k$ at all.* Sensor 2 may backoff and still transmit component $k$ regardless of Sensor 1's decision.
  - *Sensor* 2 *does not transmit.* [DONE.]
  - *Sensor* 2 *does transmit.*
- *Sensor* 2 *does not backoff component $k$ at all.* [DONE.]
- *Both sensors decide to transmit/backoff component $k$.*
  - *Sensor* 1*'s round-trip is longer than sensor* 2*'s backoff duration.* In math: $\underline{m}_1 \Delta t^{(s)} + \underline{m}'_1 \Delta t^{(b)} \leq B$.
    - *Sensor* 2 *does not transmit.* [NOT APPLICABLE.]
    - *Sensor* 2 *does transmit.*
  - *Sensor* 2*'s backoff duration is longer than sensor* 1*'s round-trip.* In math $B < \underline{m}_1 \Delta t^{(s)} + \underline{m}'_1 \Delta t^{(b)}$.
    - *Sensor* 2 *does not transmit.* [DONE.]
    - *Sensor* 2 *does transmit.*

**Lemma 9.** When $p^{(b)} = 1$, sensor 2 does not transmit if $S_2 > \Delta t^{(u)} + \Delta t^{(b)}$ and $|y_{1k}(t) - y_{2k}(t)| < \epsilon$ where $t$ is the time of sensor 1's transmission of component $k$. If the component doesn't change during the backoff duration, $|y_{1k}(t) - y_{2k}(t)| < \epsilon$ turns into $|V_1 - V_2| < \epsilon$ for $V_i \sim \mathcal{N}(0, \sigma^2)$.

Can compute expected reduction of power in shared components for OUT over IN. Since there is only at most one change per $\Delta t$ seconds, expected difference in power for shared components IN minus OUT = $q(P_u - P_D)$, where $q$ is the probability that the true environment will change in a specific interval of $\Delta t$.

**Remark 8** (General Environments and Multiple Asynchronous Sensors)**.** The analysis in this section yields insights which are applicable to environments more general than the one described in Setting 3. In particular, for more than one change and overlapping components sensed by multiple sensors, use backoff times for all packet transmissions in order to prevent collisions. The backoff strategy also works for sensors which observe the environment asynchronously. Given the triggering rule still causes changes to be detected if the environment moved $\geq \rho$ from its previous transmission, so even a continuous environment will be discretized into steps of size $\rho$. Despite possible extensions to more general scenarios, we make the assumptions in Section **??**, Setting 3, and Setting 4 in our analysis purely for pedagogical purposes. The insights we derive in this section can also be observed under these generalities, but by introducing unnecessary complexities, the same insights become less cleanly interpretable. □

**Setting 5** (White-Noise, Event-Triggered Transmissions)**.** Both sensors are corrupted with white noise of variance $\sigma > 0$. The environment evolves as the Markov chain, random-walk-like environment from Setting 3 with stepsize $C(c) \sim U[1.5, 3.5]$ distributed uniformly between $a, b \in \mathbb{R}^+$ for all $c \in \mathbb{N}$ and $q_i = p/2n$ for each $i \in \{1, \cdots, n\}$. Instead of a fixed periodic transmission scheme like Setting 2, we assume the event-triggered communication structure of Setting 4. We choose triggering strategy threshold values $\theta = \epsilon = 1.4$.

The results of the experiments over different values of $\sigma$ are shown in Figure 2.9. When we choose a value of $\mu > \Delta t$, the response changes as in Figure 2.10. Unlike the no-noise case of Section **??**, sensors may still transmit based on changes observed due to noisy observations, not true transitions of the environment state.

Intuition is more easily derived in the case of no white noise $\sigma = 0$, but also hold for when $\sigma \neq 0$. First, because both sensors operate on event-triggered transmissions, $OUT(\epsilon)$ can never beat $IN(\theta)$ in the MSE when $\sigma = 0$. Thus, any variations such that $OUT(\epsilon)$ obtains less MSE than $IN(\theta)$ (see Figure 2.10) is a

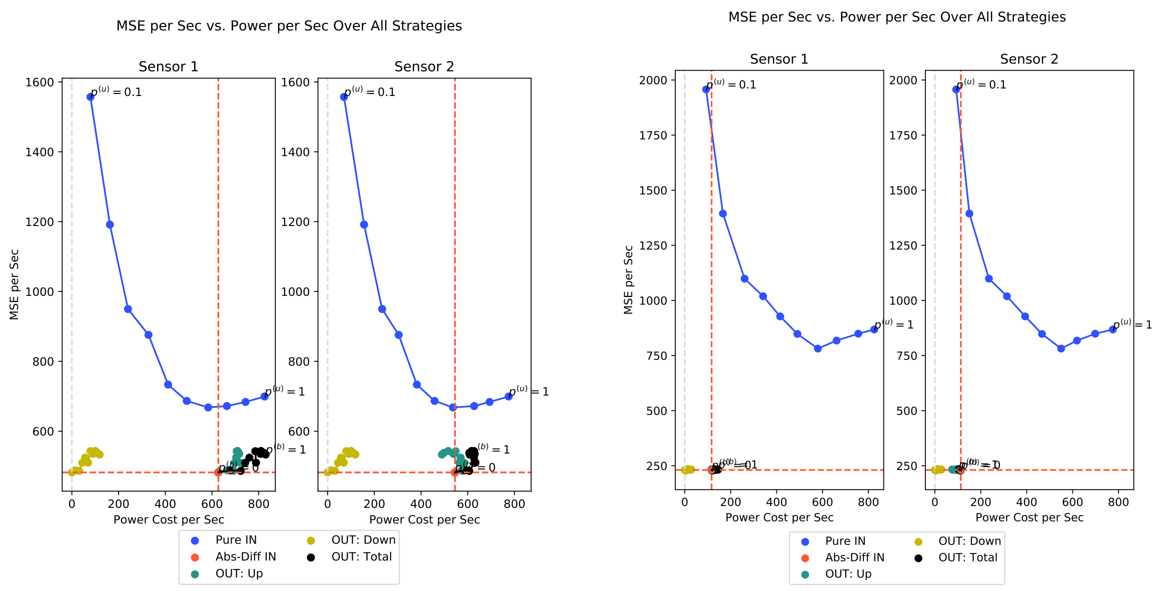

**Figure 2.9:** A version of Figure 2.5 when $\sigma \neq 0$: MSE per second vs cost of power per second for both sensors 1 and 2, averaged over 20 Monte-Carlo simulations. For the pure INformation architecture, the probability $p^{(u)}$ of transmission when the triggering condition is met is varied over $\{0.1, 0.2, \cdots, 1\}$; for the smart OUTformation architecture, the broadcast probability $p^{(b)}$ is ranged over $\{0, 0.05, 0.1, \cdots, 1\}$. For both figures, we use $\Delta t = 20, p = 0.85$, and the mean of the random backoff duration is $\mu = 10$. [Top Row] $\sigma = 0.3, a = 1.0, b = 3.0$, for large noise. [Bottom Row] $\sigma = 0.1, a = 2.0, b = 4.0$, for small noise.

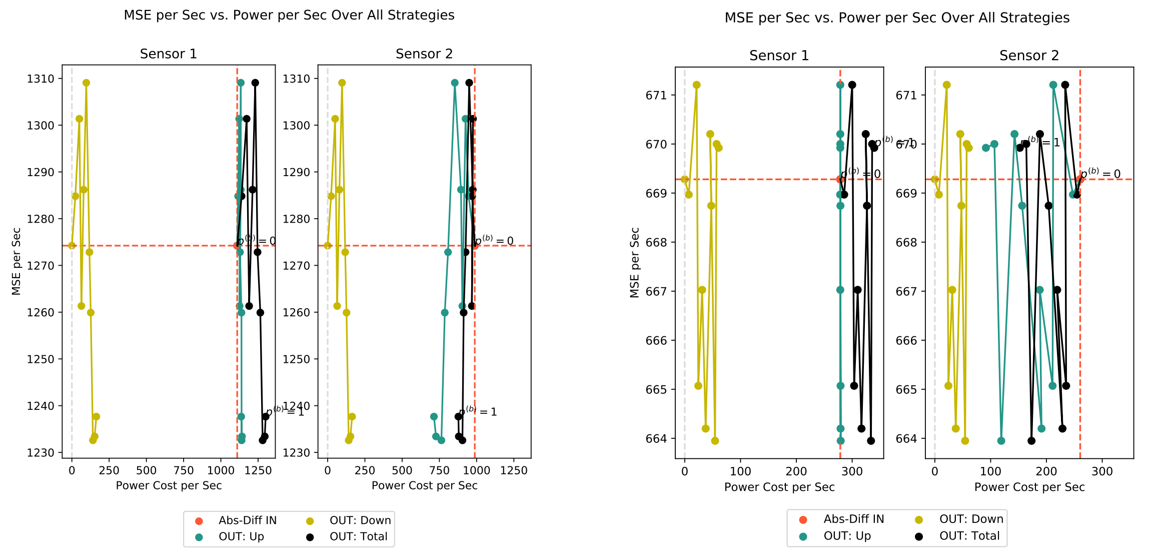

**Figure 2.10:** A version of Figure 2.9 when the mean backoff duration $\mu$ is smaller than $\Delta t$ of the Markov chain; $\mu = 10$, $\Delta t = 8$, $p = 0.85$. For clearer comparison, the pure INformation architectures are omitted. [Top Row] $\sigma = 0.3, a = 1.0, b = 3.0$, for large noise. [Bottom Row] $\sigma = 0.1, a = 2.0, b = 4.0$, for small noise.

result of random noise. Second, in all experiments, sensor 1 uses more total power as $p^{(b)} \to 1$ due to an increasing number of broadcasts. On the other hand, sensor 2 uses less total power as $p^{(b)} \to 1$ because the broadcast containing sensor 1's update terminates its backed-off transmission, which contains the same value.

When $\mu < \Delta t$ (see Figure 2.9), it is possible to use less overall power cost under $OUT(\epsilon)$ than under $IN(\theta)$, but this depends on the rate of uplink/downlink power $P_U, P_D$ expended per packet. When $\mu > \Delta t$ (see Figure 2.10), the overall MSE is larger than when $\mu > \Delta t$. This is expected because $\mu > \Delta t$ implies that on average, sensor 2's observations about $\mathbf{x}(c\Delta t)$ are received when the true state is already changed to $\mathbf{x}((c+1)\Delta t)$. Because the nature of the broadcast rule is to ignore outdated transmissions, the performance of such an $OUT(\epsilon)$ architecture becomes equivalent to an architecture which only uses sensor 1.

Overall, for environments under Setting 5, an OUTformation architecture can consistently reach near the same level of MSE performance as an INformation architecture with less power expenditure, provided $P_U << P_D$ and $\sigma$ is small relative to the magnitude of changes in the true environment. Otherwise, it is better to use an INformation architecture because unlike Setting 2, Setting 5 does not fully showcase the ability of OUTformation to shorten transmission delays by reducing the number of packets transmitted for both sensors. There are two reasons for this. First, event-triggering is used to verify satisfaction of the triggering rule with the current observation, so any environment change which is detected by one sensor is always detected by the other sensor, unless $\sigma$ is too large. Second, the random backoff strategy does not change the value of the backed-off transmission; as a result, sensor 1 performs similarly under either $IN(\theta)$ or $OUT(\epsilon)$.

## 2.5 The Distributed Kalman Consensus Filter

Old writing.

### 2.5.1 Setup

Now consider a setting where we have multiple sensors giving us multiple observations for each prediction-update iteration.

$$\begin{aligned} \mathbf{x}_{k+1} &= A_k \mathbf{x}_k + w_k \quad \text{(system)} \\ \mathbf{y}_{k,i} &= C_{k,i} \mathbf{x}_k + v_{k,i} \quad \text{(observations)} \end{aligned}$$

for sensors $i = 1, \cdots, N$. We again have $\mathbf{x}_k, w_k \in \mathbb{R}^n$ and $A_k \in \mathbb{R}^{n \times n}$, and for each sensor $i$ $\mathbf{y}_{k,i}, v_{k,i} \in \mathbb{R}^{m_i}, C_{k,i} \in \mathbb{R}^{m_i \times n}$ where $m_i \le n$. We have $w_k \sim \mathcal{N}(0, Q_k)$, and $v_{k,i}$ with mean 0 and covariance $\mathbb{E}[v_{k,i} vl, j] = R_{ij}\delta_{kl}$. Further denote the combined observations with $\mathbf{y}_k := (\mathbf{y}_{k,1}, \cdots, \mathbf{y}_{k,N}) \in \mathbb{R}^m$, $C_k := \begin{bmatrix} C_{k,1}^T & \cdots & C_{k,N}^T \end{bmatrix}^T \in \mathbb{R}^{m \times n}$, and $v = (v_{k,1}, \cdots, v_{k,N}) \in \mathbb{R}^m$ with covariance matrix $R = [R_{ij}] \in \mathbb{R}^{m \times m}$, where $m := \sum_i m_i$. We will also denote $\mathscr{A}_k = \sigma(\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_k)$ as before.

Some changes to the previous notation are listed as follows:

$$\begin{aligned} \hat{\mathbf{x}}_{k|k} &\to \hat{\mathbf{x}}_k = \hat{E}[\mathbf{x}_k | \mathscr{A}_k] \\ \hat{\mathbf{x}}_{k|k-1} &\to \hat{\mathbf{x}}_k^- = \hat{E}[\mathbf{x}_k | \mathscr{A}_{k-1}] \\ \varepsilon_k &= \mathbf{x}_k - \hat{\mathbf{x}}_k, \ \varepsilon_k^- = \mathbf{x}_k - \hat{\mathbf{x}}_k^- \\ \Sigma_{k|k} &\to \Sigma_k = \mathbb{E}[\hat{\mathbf{x}}_k \hat{\mathbf{x}}_k^T] \end{aligned}$$

$$\Sigma_{k|k-1} \to \Sigma_k^- = \mathbb{E}[\hat{\mathbf{x}}_k^- (\hat{\mathbf{x}}_k^-)^T]$$

Note that with this notation $\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_{k+1}^-$. This makes it clear that the $k-1$th iteration's posterior is used for the $k$th iteration's prior.

The original Kalman filter equations of Section 1.4.2 become:

$$\begin{aligned}
\Sigma_k &= \Sigma_k^- - L_k C_k \Sigma_k^- \\
\Sigma_k^+ &= A_k \Sigma_k^- A_k^T + Q_k \\
\hat{\mathbf{x}}_k &= \hat{\mathbf{x}}_k^- + L_k(\mathbf{y}_k - C_k \hat{\mathbf{x}}_k^-) \\
\hat{\mathbf{x}}_k^+ &= A_k \hat{\mathbf{x}}_k \\
L_k &:= \Sigma_k^- C_k^T (C_k \Sigma_k^- C_k^T + R_k)^{-1} \text{ Kalman gain}
\end{aligned}$$

We will express these equations in an alternative way that will make it easier to extend to a distributed framework. Using matrix inversion (Lemma 2), we see that:

$$\begin{aligned}
\Sigma_k^- - L_k C_k \Sigma_k^- &= \Sigma_k^- - \Sigma_k^- C_k^T (C_k \Sigma_k^- C_k^T + R_k)^{-1} C_k \Sigma_k^- = ((\Sigma_k^-)^{-1} + C_k^T R_k^{-1} C_k)^{-1} \\
&\implies \Sigma_k = ((\Sigma_k^-)^{-1} + C_k^T R_k^{-1} C_k)^{-1}
\end{aligned}$$

so that

$$\begin{aligned}
\Sigma_k &= ((\Sigma_k^-)^{-1} + C_k^T R_k^{-1} C_k)^{-1} \\
\Sigma_k^+ &= A_k \Sigma_k^- A_k^T + Q_k \\
\hat{\mathbf{x}}_k &= \hat{\mathbf{x}}_k^- + L_k(\mathbf{y}_k - C_k \hat{\mathbf{x}}_k^-) \\
\hat{\mathbf{x}}_k^+ &= A_k \hat{\mathbf{x}}_k \\
L_k &:= \Sigma_k C_k^T R_k^{-1}
\end{aligned}$$

Furthermore, define $\mathbf{z}_k = C_k^T R_k^{-1} \mathbf{y}$ to be the weighted sensor data, and $S_k = C_k^T R_k^{-1} C_k$ to be the information matrix. Our equations become

$$\begin{aligned}
\Sigma_k &= ((\Sigma_k^-)^{-1} + S_k)^{-1} \\
\Sigma_k^+ &= A_k \Sigma_k^- A_k^T + Q_k \\
\hat{\mathbf{x}}_k &= \hat{\mathbf{x}}_k^- + \Sigma_k C_k^T R_k^{-1} (\mathbf{y}_k - C_k \hat{\mathbf{x}}_k^-) \\
&= \hat{\mathbf{x}}_k^- + \Sigma_k (\mathbf{z}_k - S_k \hat{\mathbf{x}}_k^-) \\
\hat{\mathbf{x}}_k^+ &= A_k \hat{\mathbf{x}}_k \\
S_k &:= C_k^T R_k^{-1} C_k
\end{aligned}$$

### 2.5.2 Incorporating Consensus

For sensor $i$, the state estimate equation is given by:

$$\hat{x}_{k,i} = \hat{x}_{k,i}^- + L_{k,i}(\mathbf{y}_{k,i} - C_{k,i} \hat{\mathbf{x}}_{k,i}^-) + G_{k,i} \sum_{j \in \mathcal{N}_i} (\hat{\mathbf{x}}_{k,j}^- - \hat{\mathbf{x}}_{k,i}^-) \quad \textbf{(*)}$$

where $i$ denotes the index of the sensor in every subscript, $G_k$ denotes the consensus gain and is dependent upon the topology of the sensor network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, $\mathcal{N}_i := \{j : (i,j) \in \mathcal{E}\}$ is the set of neighboring sensors to sensor $i$. $\hat{x}_{k,i}$ denotes the estimate of the state $\mathbf{x}_k$ when only the observations $\mathscr{A}_{k,i} := \sigma(\mathbf{y}_{0,i}, \cdots, \mathbf{y}_{k,i})$ are taken into account.

A distributed framework that necessitates consensus among the sensors is described by Theorem 1 of [9]. The problem to be solved is to find the optimal set of Kalman gains $L_{k,i}$, sequence of estimates for both the state $\hat{\mathbf{x}}_k$ and the covariance of error $\Sigma_k$ such that we can minimize the mean-squared estimation error $\sum_i \mathbb{E}[\|\hat{\mathbf{x}}_i - \mathbf{x}\|^2]$, where the norm is with respect to the time index $k$. Note that this quantity that we want to minimize is simply the trace of the matrix $\Sigma_{k,i}$.

Define the edge-covariance matrices as follows:

$$\Sigma_{k,ij} = \mathbb{E}[\varepsilon_{k,i}\varepsilon_{k,j}^T], \ \ \Sigma_{k,ij}^- = \mathbb{E}[\varepsilon_{k,i}^-(\varepsilon_{k,j}^-)^T]$$

Consider two neighboring nodes $i$ and $j$ with respective estimation errors $\varepsilon_{k,i} = \hat{\mathbf{x}}_{k,i} - \mathbf{x}_k$ and $\varepsilon_{k,j} = \hat{\mathbf{x}}_{k,j} - \mathbf{x}_k$. Define $F_{k,i} := I - L_{k,i}C_{k,i}$. We will begin with the state estimation equation **(*)** for each of $i$ and $j$, and subtract both sides of them by $\mathbf{x}_k$:

$$\begin{aligned}
\varepsilon_{k,i} &= \varepsilon_{k,i}^- + L_{k,i}(\mathbf{y}_{k,i} - C_{k,i}\hat{\mathbf{x}}_{k,i}) + G_{k,i}\sum_{r\in\mathcal{N}_i}(\hat{\mathbf{x}}_{k,r}^- - \hat{\mathbf{x}}_{k,i}^-) \\
&= F_{k,i}\varepsilon_{k,i}^- + L_{k,i}C_{k,i}\varepsilon_{k,i}^- + L_{k,i}(\mathbf{y}_{k,i} - C_{k,i}\hat{\mathbf{x}}_{k,i}) + G_{k,i}\sum_{r\in\mathcal{N}_i}(\hat{\mathbf{x}}_{k,r}^- - \hat{\mathbf{x}}_{k,i}^-) \\
&= F_{k,i}\varepsilon_{k,i}^- + L_{k,i}\underbrace{(\mathbf{y}_{k,i} - C_{k,i}\mathbf{x}_{k,i})}_{v_{k,i}} + G_{k,i}\sum_{r\in\mathcal{N}_i}(\hat{\mathbf{x}}_{k,r}^- - \hat{\mathbf{x}}_{k,i}^-)
\end{aligned}$$

and similarly:

$$\varepsilon_{k,j} = F_{k,j}\varepsilon_{k,j}^- + L_{k,j}v_{k,j} + G_{k,j}\sum_{s\in\mathcal{N}_i}(\hat{\mathbf{x}}_{k,s}^- - \hat{\mathbf{x}}_{k,j}^-)$$

Substituting the two equations above into our definition of the edge-covariance:

$$\begin{aligned}
\Sigma_{k,ij} &= \mathbb{E}[\varepsilon_{k,i}\varepsilon_{k,j}^T] \\
&= \mathbb{E}[F_{k,i}\varepsilon_{k,i}^-(\varepsilon_{k,j}^-)^T F_{k,j}^T] + \mathbb{E}\left[F_{k,i}\varepsilon_{k,i}^-\sum_{s\in\mathcal{N}_j}(\hat{\mathbf{x}}_{k,s}^- - \hat{\mathbf{x}}_{k,j}^-)^T G_{k,j}^T\right] + \mathbb{E}[L_{k,i}v_{k,i}v_{k,j}^T L_{k,j}^T] \\
&\quad + \mathbb{E}\left[G_{k,i}\varepsilon_{k,i}^-\sum_{r\in\mathcal{N}_i}(\hat{\mathbf{x}}_{k,r}^- - \hat{\mathbf{x}}_{k,i}^-)^T F_{k,j}^T\right] + \mathbb{E}\left[G_{k,i}\sum_{r\in\mathcal{N}_i}(\hat{\mathbf{x}}_{k,r}^- - \hat{\mathbf{x}}_{k,i}^-)\sum_{s\in\mathcal{N}_i}(\hat{\mathbf{x}}_{k,s}^- - \hat{\mathbf{x}}_{k,j}^-)^T G_{k,j}^T\right] \\
&= F_{k,i}\Sigma_{k,ij}^- F_{k,j}^T + F_{k,i}\sum_{s\in\mathcal{N}_j}\mathbb{E}[\varepsilon_{k,i}^-(\hat{\mathbf{x}}_{k,s}^- - \hat{\mathbf{x}}_{k,j}^-)^T]G_{k,j}^T + L_{k,i}R_{k,ij}L_{k,j}^T \\
&\quad + G_{k,i}\sum_{r\in\mathcal{N}_i}\mathbb{E}[(\hat{\mathbf{x}}_{k,r}^- - \hat{\mathbf{x}}_{k,i}^-)(\varepsilon_{k,j}^-)^T]F_{k,j}^T + G_{k,i}\sum_{r\in\mathcal{N}_i}\sum_{s\in\mathcal{N}_j}\mathbb{E}[(\hat{\mathbf{x}}_{k,r}^- - \hat{\mathbf{x}}_{k,i}^-)(\hat{\mathbf{x}}_{k,s}^- - \hat{\mathbf{x}}_{k,j}^-)^T]G_{k,j}^T
\end{aligned}$$

Expanding each of the expected value quantities yields:

$$\sum_{s\in\mathcal{N}_j}\mathbb{E}[\varepsilon_{k,i}^-(\hat{\mathbf{x}}_{k,s}^- - \hat{\mathbf{x}}_{k,j}^-)^T] = \sum_{s\in\mathcal{N}_j}\mathbb{E}[\varepsilon_{k,i}^-(\varepsilon_{k,s}^- - \varepsilon_{k,j}^-)^T] \text{ by } \pm\mathbf{x}_k$$

$$= \sum_{s\in\mathcal{N}_j} \Sigma^-_{k,is} - \Sigma^-_{k,ij}$$

$$\sum_{r\in\mathcal{N}_i} \mathbb{E}[(\hat{\mathbf{x}}^-_{k,r} - \hat{\mathbf{x}}^-_{k,i})(\varepsilon^-_{k,j})^T] = \sum_{r\in\mathcal{N}_i} \Sigma^-_{k,rj} - \Sigma^-_{k,ij}$$

and we will denote

$$D_{k,ij} := \sum_{r\in\mathcal{N}_i}\sum_{s\in\mathcal{N}_j} \mathbb{E}[(\hat{\mathbf{x}}^-_{k,r} - \hat{\mathbf{x}}^-_{k,i})(\hat{\mathbf{x}}^-_{k,s} - \hat{\mathbf{x}}^-_{k,j})^T] = \sum_{r\in\mathcal{N}_i}\sum_{s\in\mathcal{N}_j} (\Sigma^-_{k,rs} - \Sigma^-_{k,rj} - \Sigma^-_{k,is} + \Sigma^-_{k,ij})$$

which yields the expression:

$$\Sigma_{k,ij} = F_{k,i}\Sigma^-_{k,ij}F^T_{k,j} + F_{k,i}\left(\sum_{s\in\mathcal{N}_j} \Sigma^-_{k,is} - \Sigma^-_{k,ij}\right) G^T_{k,j} + L_{k,i}R_{k,ij}L^T_{k,j} + G_{k,i}\left(\sum_{r\in\mathcal{N}_i} \Sigma^-_{k,rj} - \Sigma^-_{k,ij}\right) F^T_{k,j} \\ + G_{k,i}D_{k,ij}G^T_{k,j}$$

Now let us optimize over the trace of $\Sigma_{k,i}$. From the formula above, we have:

$$\Sigma_{k,i} = F_{k,i}\Sigma^-_{k,i}F^T_{k,i} + F_{k,i}\left(\sum_{s\in\mathcal{N}_i} \Sigma^-_{k,is} - \Sigma^-_{k,i}\right) G^T_{k,i} + L_{k,i}R_{k,i}L^T_{k,i} + G_{k,i}\left(\sum_{r\in\mathcal{N}_i} \Sigma^-_{k,ri} - \Sigma^-_{k,i}\right) F^T_{k,i} \\ + G_{k,i}D_{k,i}G^T_{k,i}$$

where $D_{k,i} = \sum_{r\in\mathcal{N}_i}\sum_{s\in\mathcal{N}_i}(\Sigma^-_{k,rs} - \Sigma^-_{k,ri} - \Sigma^-_{k,is} + \Sigma^-_{k,i})$.

We will take the derivative and set it equal to 0, then determine the minimizing Kalman gain $L_{k,i}$. Note that $F_{k,i} = I - L_{k,i}C_{k,i}$ is dependent on $L_{k,i}$.

$$\frac{\partial \mathrm{tr}(\Sigma_{k,i})}{\partial L_{k,i}} = \frac{\partial \mathrm{tr}(F_{k,i}\Sigma^-_{k,i}F^T_{k,i})}{\partial L_{k,i}} + \frac{\partial \mathrm{tr}\left[F_{k,i}\left(\sum_{s\in\mathcal{N}_i} \Sigma^-_{k,is} - \Sigma^-_{k,i}\right) G^T_{k,i}\right]}{\partial L_{k,i}} + \frac{\partial \mathrm{tr}\left(L_{k,i}R_{k,i}L^T_{k,i}\right)}{\partial L_{k,i}} \\ + \frac{\partial \mathrm{tr}\left[G_{k,i}\left(\sum_{r\in\mathcal{N}_i} \Sigma^-_{k,ri} - \Sigma^-_{k,i}\right) F^T_{k,i}\right]}{\partial L_{k,i}} := 0$$

Recall the following matrix derivative properties:

$$\frac{\partial \mathrm{tr}(XA)}{\partial X} = \frac{\partial \mathrm{tr}(AX)}{\partial X} = A^T, \quad \frac{\partial \mathrm{tr}(AX^T)}{\partial X} = A$$
$$\frac{\partial \mathrm{tr}(XAX^T)}{\partial X} = X(A + A^T)$$

where $A$ and $X$ are two matrices of any dimensions such that the multiplication makes sense.

For each term in the sum above:

$$\frac{\partial \mathrm{tr}(F_{k,i}\Sigma^-_{k,i}F^T_{k,i})}{\partial L_{k,i}} = \frac{\partial \mathrm{tr}((I - L_{k,i}C_{k,i})\Sigma^-_{k,i}(I - L_{k,i}C_{k,i})^T)}{\partial L_{k,i}}$$

$$= \frac{\partial \mathrm{tr}(-L_{k,i}C_{k,i}\Sigma^-_{k,i} - \Sigma^-_{k,i}C^T_{k,i}L^T_{k,i} + L_{k,i}C_{k,i}\Sigma^-_{k,i}C^T_{k,i}L^T_{k,i})}{\partial L_{k,i}}$$
$$= -2\Sigma^-_{k,i}C^T_{k,i} + 2L_{k,i}C_{k,i}\Sigma^-_{k,i}C^T_{k,i}$$

$$\frac{\partial \mathrm{tr}\left[F_{k,i}\left(\sum_{s\in\mathcal{N}_i}\Sigma^-_{k,is} - \Sigma^-_{k,i}\right)G^T_{k,i}\right]}{\partial L_{k,i}} = \frac{\partial \mathrm{tr}\left[(I - L_{k,i}C_{k,i})\left(\sum_{s\in\mathcal{N}_i}\Sigma^-_{k,is} - \Sigma^-_{k,i}\right)G^T_{k,i}\right]}{\partial L_{k,i}}$$
$$= -G_{k,i}\left(\sum_{s\in\mathcal{N}_i}\Sigma^-_{k,is} - \Sigma^-_{k,i}\right)C^T_{k,i}$$

$$\frac{\partial \mathrm{tr}\left(L_{k,i}R_{k,i}L^T_{k,i}\right)}{\partial L_{k,i}} = 2L_{k,i}R_{k,i}$$

$$\frac{\partial \mathrm{tr}\left[G_{k,i}\left(\sum_{r\in\mathcal{N}_i}\Sigma^-_{k,ri} - \Sigma^-_{k,i}\right)F^T_{k,i}\right]}{\partial L_{k,i}} = \frac{\partial \mathrm{tr}\left[G_{k,i}\left(\sum_{r\in\mathcal{N}_i}\Sigma^-_{k,ri} - \Sigma^-_{k,i}\right)(I - L_{k,i}C_{k,i})^T\right]}{\partial L_{k,i}}$$
$$= -G_{k,i}\left(\sum_{r\in\mathcal{N}_i}\Sigma^-_{k,ri} - \Sigma^-_{k,i}\right)C^T_{k,i}$$

and overall, we have:

$$-2\Sigma^-_{k,i}C^T_{k,i} + 2L_{k,i}C_{k,i}\Sigma^-_{k,i}C^T_{k,i} - 2G_{k,i}\left(\sum_{s\in\mathcal{N}_i}\Sigma^-_{k,is} - \Sigma^-_{k,i}\right)C^T_{k,i} + 2L_{k,i}R_{k,i} = 0$$
$$\Longrightarrow L_{k,i}\left(C_{k,i}\Sigma^-_{k,i}C^T_{k,i} + R_{k,i}\right) = \Sigma^-_{k,i}C^T_{k,i} + G_{k,i}\left(\sum_{s\in\mathcal{N}_i}\Sigma^-_{k,is} - \Sigma^-_{k,i}\right)C^T_{k,i}$$

Our optimal Kalman gain is:

$$L_{k,i} = \left(\Sigma^-_{k,i}C^T_{k,i} + G_{k,i}\left(\sum_{s\in\mathcal{N}_i}\Sigma^-_{k,is} - \Sigma^-_{k,i}\right)C^T_{k,i}\right)\left(C_{k,i}\Sigma^-_{k,i}C^T_{k,i} + R_{k,i}\right)^{-1}$$

Finally, the update rule for the state can be determined as follows. Subtract the system dynamics $\mathbf{x}_{k+1} = A_k\mathbf{x}_k + w_k$ with the Kalman filter update equation $\hat{\mathbf{x}}^+_{k,i} = A_k\hat{\mathbf{x}}_{k,i}$ to get:

$$\varepsilon^+_{k,i} = A_k\varepsilon_{k,i} + w_k$$

and likewise for agent $j$, $\varepsilon^+_{k,j} = A_k\varepsilon_{k,j} + w_k$. For the covariance of error update:

$$\Sigma^+_{k,ij} = \mathbb{E}[\varepsilon^+_{k,i}(\varepsilon^+_{k,j})^T] = A_k\Sigma_{k,ij}A^T_k + Q_k$$

The full update equations for the Kalman-Consensus filter are:

$$\hat{x}_{k,i} = \hat{x}^-_{k,i} + L_{k,i}(\mathbf{y}_{k,i} - C_{k,i}\hat{\mathbf{x}}^-_{k,i}) + G_{k,i}\sum_{j\in\mathcal{N}_i}(\hat{\mathbf{x}}^-_{k,j} - \hat{\mathbf{x}}^-_{k,i})$$

$$
\begin{aligned}
\Sigma_{k,ij} &= F_{k,i}\Sigma^-_{k,ij}F^T_{k,j} + F_{k,i}\left(\sum_{s\in\mathcal{N}_j}\Sigma^-_{k,is} - \Sigma^-_{k,ij}\right)G^T_{k,j} + L_{k,i}R_{k,ij}L^T_{k,j} + G_{k,i}\left(\sum_{r\in\mathcal{N}_i}\Sigma^-_{k,rj} - \Sigma^-_{k,ij}\right)F^T_{k,j} \\
&\qquad + G_{k,i}D_{k,ij}G^T_{k,j} \\
L_{k,i} &= \left(\Sigma^-_{k,i}C^T_{k,i} + G_{k,i}\left(\sum_{s\in\mathcal{N}_i}\Sigma^-_{k,is} - \Sigma^-_{k,i}\right)C^T_{k,i}\right)\left(C_{k,i}\Sigma^-_{k,i}C^T_{k,i} + R_{k,i}\right)^{-1} \\
\Sigma^+_{k,ij} &= A_k\Sigma_{k,ij}A^T_k + Q_k \\
\hat{\mathbf{x}}^+_{k,i} &= A_k\hat{\mathbf{x}}_{k,i}
\end{aligned}
$$

## 2.6 The Multitarget Moment Density Filter

### 2.6.1 Derivation of the MMDF

# Bibliography

[1] B. Øksendal, *Stochastic Differential Equations: An Introduction with Applications*, ser. Universitext. Berlin, Germany: Springer Berlin, 2010. [Online]. Available: https://books.google.com/books?id=EQZEAAAAQBAJ

[2] B. Hajek, "Notes for ECE534: An exploration of random processes for engineers," 2006. [Online]. Available: http://www.ifp.uiuc.edu/~hajek/Papers/randomprocJuly06.pdf

[3] R. P. S. Mahler, "Multitarget bayes filtering via first-order multitarget moments," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 39, no. 4, pp. 1152–1178, 2003.

[4] M. Liggins, C.-Y. Chong, I. Kadar, M. Alford, V. Vannicola, and S. Thomopoulos, "Distributed fusion architectures and algorithms for target tracking," *Proceedings of the IEEE*, vol. 85, no. 1, pp. 95–107, 1997.

[5] F. Castanedo, "A review of data fusion techniques," *The Scientific World Journal*, vol. 2013, pp. 1–19, Oct 2013.

[6] B. Chen, W.-A. Zhang, L. Yu, G. Hu, and H. Song, "Distributed fusion estimation with communication bandwidth constraints," *IEEE Transactions on Automatic Control*, vol. 60, no. 5, pp. 1398–1403, 2015.

[7] C.-Y. Chong, "Hierarchical estimation," in *Proceedings of 2nd MIT/ONR Workshop on Distributed Information and Decision Systems Motivated by Naval Command Control and Communication (C3) Problems*, Jul 1979, pp. 205–220.

[8] S. Han and S.-J. Chung, "Incremental nonlinear stability analysis for stochastic systems perturbed by Lévy noise," *International Journal of Robust and Nonlinear Control*, Oct 2021, submitted.

[9] R. Olfati-Saber, "Kalman-consensus filter : Optimality, stability, and performance," in *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, Dec 2009, pp. 7036–7042.