Sliding Window Filtering
— Batch Estimation Using a Subset of Data —

Charles C. Cossette and Prof. James Richard Forbes

McGill University, Department of Mechanical Engineering

November 7, 2022

Problem Statement

- The batch state estimation framework is a robust, accurate state estimation technique.
- However, as a robot moves in time, states cannot be added into the batch estimation problem endlessly.
- The complexity of the state estimation task would grow with the life of the robot.
- A version of the batch estimation problem that has constant complexity is needed.
- This is the *sliding window filter*.
- Again, the following process and measurement models

$$
\begin{aligned}
\mathbf{x}_k &= \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{w}_{k-1}), \\
\mathbf{y}_k &= \mathbf{g}(\mathbf{x}_k, \mathbf{v}_k),
\end{aligned}
$$

  will be used, where $\mathbf{w}_{k-1}, \mathbf{v}_k$ are zero-mean Gaussian noise.

Scenario

- Suppose a robot starts at time $k = 0$. It travels for K discrete time steps until it reaches time k_1.

$$\underbrace{\mathbf{x}_0 \quad \mathbf{x}_1 \qquad \dots \qquad \dots \qquad \dots \qquad \mathbf{x}_{k_1}}_{\text{perform full batch estimate}}$$

Scenario

- Suppose a robot starts at time $k = 0$. It travels for K discrete time steps until it reaches time k_1.

$$\underbrace{\mathbf{x}_0 \quad \mathbf{x}_1 \quad \dots \quad \dots \quad \dots \quad \mathbf{x}_{k_1}}_{\text{perform full batch estimate}}$$

- The robot then continues to travel to time k_2.

$$\underbrace{\mathbf{x}_0 \quad \mathbf{x}_1 \quad \dots \quad \dots \quad \dots \quad \mathbf{x}_{k_1}}_{\text{perform full batch estimate}} \quad \mathbf{x}_{k_1+1} \quad \dots \quad \mathbf{x}_{k_2}$$

Scenario

- Suppose a robot starts at time $k = 0$. It travels for K discrete time steps until it reaches time k_1.

$$\underbrace{\mathbf{x}_0 \quad \mathbf{x}_1 \quad \dots \quad \dots \quad \dots \quad \mathbf{x}_{k_1}}_{\text{perform full batch estimate}}$$

- The robot then continues to travel to time k_2.

$$\underbrace{\mathbf{x}_0 \quad \mathbf{x}_1 \quad \dots \quad \dots \quad \dots \quad \mathbf{x}_{k_1}}_{\text{perform full batch estimate}} \quad \mathbf{x}_{k_1+1} \quad \dots \quad \mathbf{x}_{k_2}$$

- The robot then removes the m oldest states from its active state vector, and performs a new batch estimate.

$$\underbrace{\mathbf{x}_0 \quad \mathbf{x}_1 \quad \dots \quad \mathbf{x}_{m-1} \quad \mathrlap{\overbrace{\mathbf{x}_m \quad \dots \quad \mathbf{x}_{k_1} \quad \mathbf{x}_{k_1+1} \quad \dots \quad \mathbf{x}_{k_2}}^{\text{new window of length } K}} \phantom{\mathbf{x}_m \quad \dots \quad \mathbf{x}_{k_1}}}_{\text{old window of length } K}$$

Marginalization of the Old States

- However, we should not simply “delete” the oldest states.
- It is more appropriate to *marginalize* them out.

Definition (Marginalization)

Recall that *marginalization* refers to integrating a joint PDF $p(\mathbf{x}, \mathbf{y})$ with respect to some of the variables, such as $\mathbf{x}$

$$\int_{-\infty}^{\infty} p(\mathbf{x}, \mathbf{y}) \mathrm{d}\mathbf{x} = \int_{-\infty}^{\infty} p(\mathbf{x}|\mathbf{y}) p(\mathbf{y}) \mathrm{d}\mathbf{x} = p(\mathbf{y}) \underbrace{\int_{-\infty}^{\infty} p(\mathbf{x}|\mathbf{y}) \mathrm{d}\mathbf{x}}_{=1} = p(\mathbf{y}). \tag{1}$$

Marginalization of the Old States

$$\underbrace{\mathbf{x}_0\;\mathbf{x}_1\;\cdots\;\mathbf{x}_{m-1}\;\rlap{$\overbrace{\phantom{\mathbf{x}_m\;\cdots\;\mathbf{x}_{k_1}\;\mathbf{x}_{k_1+1}\;\cdots\;\mathbf{x}_{k_2}}}^{\text{new window of length } K}$}\mathbf{x}_m\;\cdots\;\mathbf{x}_{k_1}}_{\text{old window of length } K}\;\mathbf{x}_{k_1+1}\;\cdots\;\mathbf{x}_{k_2}$$

Using the colon notation,

- $\mathbf{x}_{0:m-1}$ are the states to be **marginalized**,
- $\mathbf{x}_{m:k_1}$ are the states that **remain** in the window, and
- $\mathbf{x}_{m:k_2}$ are the states in the **new window**.

Marginalization of the Old States

- We will start with the full batch MAP estimation problem,

$$\hat{\mathbf{x}}_{0:k_2} = \arg\max_{\mathbf{x}_{0:k_2}} p(\mathbf{x}_{0:k_2}|\check{\mathbf{x}}_0, \mathbf{u}, \mathbf{y}) \tag{2}$$

  where $\mathbf{x}_{0:k_2} = \{\mathbf{x}_0, \ldots, \mathbf{x}_{k_2}\}$.

- The full joint PDF can be expanded into factors as follows

$$\begin{aligned} p(\mathbf{x}_{0:k_2}|\check{\mathbf{x}}_0, \mathbf{u}, \mathbf{y}) &= \alpha p(\mathbf{y}_{m:k_2}|\mathbf{x}, \check{\mathbf{x}}_0, \mathbf{u}, \mathbf{y}_{1:m-1}) p(\mathbf{x}_{0:k_2}|\check{\mathbf{x}}_0, \mathbf{u}, \mathbf{y}_{1:m-1}) \\ &= \alpha p(\mathbf{y}_{m:k_2}|\mathbf{x}) p(\mathbf{x}_{0:k_2}|\check{\mathbf{x}}_0, \mathbf{u}, \mathbf{y}_{1:m-1}) \\ &= \alpha p(\mathbf{y}_{m:k_2}|\mathbf{x}) p(\mathbf{x}_{m:k_2}|\check{\mathbf{x}}_0, \mathbf{u}, \mathbf{y}_{1:m-1}, \mathbf{x}_m) \\ &\qquad \times p(\mathbf{x}_{0:m-1}|\check{\mathbf{x}}_0, \mathbf{u}, \mathbf{y}_{1:m-1}) \\ &= \alpha \left(\prod_{k=m}^{k_2} p(\mathbf{y}_k|\mathbf{x}_k) \right) \left(\prod_{k=m+1}^{k_2} p(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{u}_{k-1}) \right) \\ &\qquad \times p(\mathbf{x}_m|\check{\mathbf{x}}_0, \mathbf{u}, \mathbf{y}_{1:m-1}, \mathbf{x}_{0:m-1}) p(\mathbf{x}_{0:m-1}|\check{\mathbf{x}}_0, \mathbf{u}, \mathbf{y}_{1:m-1}). \end{aligned}$$

Marginalization of the Old States

- We may now marginalize out the oldest states by integrating with respect to $\mathbf{x}_{0:m-1}$

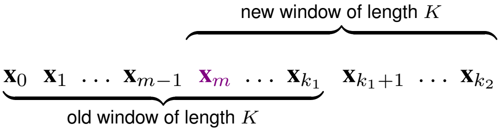
$$\int_{-\infty}^{\infty} p(\mathbf{x}_{0:k_2}|\check{\mathbf{x}}_0, \mathbf{u}, \mathbf{y})\mathrm{d}\mathbf{x}_{0:m-1} = \alpha \left(\prod_{k=m}^{k_2} p(\mathbf{y}_k|\mathbf{x}_k) \right) \left(\prod_{k=m+1}^{k_2} p(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{u}_{k-1}) \right) \\ \times \int_{-\infty}^{\infty} p(\mathbf{x}_m|\check{\mathbf{x}}_0, \mathbf{u}, \mathbf{y}_{0:m-1}, \mathbf{x}_{0:m-1}) p(\mathbf{x}_{0:m-1}|\check{\mathbf{x}}_0, \mathbf{u}, \mathbf{y}_{0:m-1})\mathrm{d}\mathbf{x}_{0:m-1} \quad (3)$$

$$p(\mathbf{x}_{m:k_2}|\check{\mathbf{x}}_0, \mathbf{u}, \mathbf{y}) = \alpha \overbrace{\left(\prod_{k=m}^{k_2} p(\mathbf{y}_k|\mathbf{x}_k) \right)}^{\text{measurements}} \overbrace{\left(\prod_{k=m+1}^{k_2} p(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{u}_{k-1}) \right)}^{\text{process model}} \\ \times \underbrace{p(\mathbf{x}_m|\check{\mathbf{x}}_0, \mathbf{u}_{0:m-1}, \mathbf{y}_{0:m-1})}_{\text{new “prior”}}. \quad (4)$$

- As with the batch MAP approach, we could now attempt to maximize (4), which would lead to a least-squares problem.

Determining the New Prior Distribution

- We are only missing one thing to set up our least-squares problem, which is $p(\mathbf{x}_m | \check{\mathbf{x}}_0, \mathbf{u}_{0:m-1}, \mathbf{y}_{0:m-1})$


new window of length K
$\mathbf{x}_0 \quad \mathbf{x}_1 \quad \dots \quad \mathbf{x}_{m-1} \quad \mathbf{x}_m \quad \dots \quad \mathbf{x}_{k_1} \quad \mathbf{x}_{k_1+1} \quad \dots \quad \mathbf{x}_{k_2}$
old window of length K

- **That is, we are looking for the distribution of $\mathbf{x}_m$ given all the measurements that occurred before it.**
- $p(\mathbf{x}_m | \check{\mathbf{x}}_0, \mathbf{u}_{0:m-1}, \mathbf{y}_{0:m-1})$ takes the role of the new “prior”, which was $p(\mathbf{x}_0 | \check{\mathbf{x}}_0)$ in the full batch scenario.

Determining the New Prior Distribution

Theorem (Marginalization)

Given the joint Gaussian probability density function

$$p(\mathbf{x}, \mathbf{y}) = \mathcal{N}\left(\left[\begin{array}{c} \boldsymbol{\mu}_x \\ \boldsymbol{\mu}_y \end{array}\right], \left[\begin{array}{cc} \boldsymbol{\Sigma}_{xx} & \boldsymbol{\Sigma}_{xy} \\ \boldsymbol{\Sigma}_{yx} & \boldsymbol{\Sigma}_{yy} \end{array}\right]\right)$$

the marginal PDF $p(\mathbf{x}) = \int_{-\infty}^{\infty} p(\mathbf{x}, \mathbf{y}) \mathrm{d}\mathbf{y}$ is given by

$$p(\mathbf{x}) = \mathcal{N}(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_{xx}). \tag{5}$$

Determining the New Prior Distribution

- Therefore, we can use our old estimates (from the previous window) $\{\hat{\mathbf{x}}_{0:m-1}, \hat{\mathbf{x}}_m\} = \hat{\mathbf{x}}_{0:m}$ to construct

$$p(\mathbf{x}_{0:m}|\check{\mathbf{x}}_0, \mathbf{u}_{0:m-1}, \mathbf{y}_{0:m-1}) = \beta \exp(-\frac{1}{2}\mathbf{e}_m(\mathbf{x}_{0:m})^\mathsf{T}\mathbf{W}_m\mathbf{e}_m(\mathbf{x}_{0:m})), \tag{6}$$

where

$$\mathbf{e}_m(\mathbf{x}_{0:m}) = \begin{bmatrix} \mathbf{x}_0 - \check{\mathbf{x}}_0 \\ \mathbf{x}_1 - \mathbf{f}(\mathbf{x}_0, \mathbf{u}_0, \mathbf{0}) \\ \vdots \\ \mathbf{x}_m - \mathbf{f}(\mathbf{x}_{m-1}, \mathbf{u}_{m-1}, \mathbf{0}) \\ \mathbf{y}_0 - \mathbf{g}(\mathbf{x}_0, \mathbf{0}) \\ \vdots \\ \mathbf{y}_{m-1} - \mathbf{g}(\mathbf{x}_{m-1}, \mathbf{0}) \end{bmatrix}, \tag{7}$$

$$\mathbf{W}_m = \mathrm{diag}(\mathbf{P}_0^{-1}, \mathbf{Q}_1^{-1}, \ldots, \mathbf{Q}_m^{-1}, \mathbf{R}_0^{-1}, \ldots, \mathbf{R}_{m-1}^{-1}). \tag{8}$$

- Although this is not Gaussian, it can be **approximated** as one by linearizing $\mathbf{e}_m(\mathbf{x}_{0:m})$.

Watch out.

- **Very important:** $\mathbf{e}_m \neq \mathbf{e}$.
- $\mathbf{e}_m$ is a “mini”/smaller vector that only contains errors involving the states being marginalized.
- You **cannot reuse the same** $\mathbf{e}, \mathbf{H}, \mathbf{W}$ matrices that were involved in the initial batch estimate.

Determining the New Prior Distribution

- The mean and covariance of a Gaussian approximation to (6) are given by

$$\boldsymbol{\mu}_{0:m} = \begin{bmatrix} \boldsymbol{\mu}_{0:m-1} \\ \boldsymbol{\mu}_m \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{x}}_{0:m-1} \\ \hat{\mathbf{x}}_m \end{bmatrix} - (\mathbf{H}_m^{\mathsf{T}}\mathbf{W}_m\mathbf{H}_m)^{-1}\mathbf{H}_m^{\mathsf{T}}\mathbf{W}_m\bar{\mathbf{e}}_m, \tag{9}$$

$$\boldsymbol{\Sigma}_{0:m} = \begin{bmatrix} \boldsymbol{\Sigma}_{0:m-1} & \boldsymbol{\Sigma}_{0:m-1,m} \\ \boldsymbol{\Sigma}_{m,0:m-1} & \boldsymbol{\Sigma}_m \end{bmatrix} = (\mathbf{H}_m^{\mathsf{T}}\mathbf{W}_m\mathbf{H}_m)^{-1}, \tag{10}$$

  where

$$\bar{\mathbf{e}}_m = \mathbf{e}_m(\hat{\mathbf{x}}_{0:m}), \qquad \mathbf{H}_m = \left.\frac{\partial \mathbf{e}_m(\mathbf{x})}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}_{0:m}}. \tag{11}$$

- This can finally be used to approximate $p(\mathbf{x}_m|\check{\mathbf{x}}_0, \mathbf{u}_{0:m-1}, \mathbf{y}_{0:m-1})$ as

$$p(\mathbf{x}_m|\check{\mathbf{x}}_0, \mathbf{u}_{0:m-1}, \mathbf{y}_{0:m-1}) \approx \mathcal{N}(\boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m). \tag{12}$$

- This is the **only approximation made** in going from the batch estimate to the sliding window filter.
- **Important: $\mathbf{e}_m, \mathbf{H}_m, \mathbf{W}_m$ are different from $\mathbf{e}, \mathbf{H}, \mathbf{W}$.**

State Estimate of the New Window

- Returning to the actual estimation, we can find the states which maximize $p(\mathbf{x}_m|\check{\mathbf{x}}_0, \mathbf{u}_{0:m-1}, \mathbf{y}_{0:m-1}) \approx \mathcal{N}(\boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)$ as the prior,

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} \alpha \left(\prod_{k=m}^{k_2} p(\mathbf{y}_k|\mathbf{x}_k) \right) \left(\prod_{k=m+1}^{k_2} p(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{u}_k) \right) p(\mathbf{x}_m|\check{\mathbf{x}}_0, \mathbf{u}_{0:m-1}, \mathbf{y}_{0:m-1}). \tag{13}$$

- We proceed as with the batch MAP framework by minimizing the negative logarithm of (13), which leads to the following nonlinear weighted least-squares problem . . .

State Estimate of the New Window

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \frac{1}{2}\mathbf{e}(\mathbf{x})\mathbf{W}\mathbf{e}(\mathbf{x}) \tag{14}$$

where

$$\mathbf{e}(\mathbf{x}) = \begin{bmatrix} \mathbf{x}_m - \boldsymbol{\mu}_m \\ \mathbf{x}_{m+1} - \mathbf{f}(\mathbf{x}_m, \mathbf{u}_m, \mathbf{0}) \\ \vdots \\ \mathbf{x}_{k_2} - \mathbf{f}(\mathbf{x}_{k_2-1}, \mathbf{u}_{k_2-1}, \mathbf{0}) \\ \mathbf{y}_m - \mathbf{g}(\mathbf{x}_m, \mathbf{0}) \\ \vdots \\ \mathbf{y}_{k_2} - \mathbf{g}(\mathbf{x}_{k_2}, \mathbf{0}) \end{bmatrix} \tag{15}$$

$$\mathbf{W} = \operatorname{diag}(\boldsymbol{\Sigma}_m^{-1}, \mathbf{Q}_{m+1}^{-1}, \ldots, \mathbf{Q}_{k_2}^{-1}, \mathbf{R}_m^{-1}, \ldots, \mathbf{R}_{k_2}^{-1}) \tag{16}$$

- This is solved as usual with the Gauss-Newton algorithm.

Summary

Sliding Window Filter

To estimate the states in the window at time k_2, denoted $\hat{\mathbf{x}}_{k_2}$ starting with the estimate of the states of the previous window at time k_1, denoted $\hat{\mathbf{x}}_{k_1}$:

1. split the previous window's estimate into the marginalized states and the remaining states

$$\hat{\mathbf{x}}_{0:k_1} = \begin{bmatrix} \hat{\mathbf{x}}_{0:m-1} \\ \hat{\mathbf{x}}_{m:k_1} \end{bmatrix}; \tag{17}$$

2. solve for $\boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m$ using (9), (10), (11);
3. construct the nonlinear least squares problem using (14), (15), (16);
4. solve the nonlinear least squares problem using the Gauss-Newton algorithm.

Sliding Window Filter vs. Extended Kalman Filter

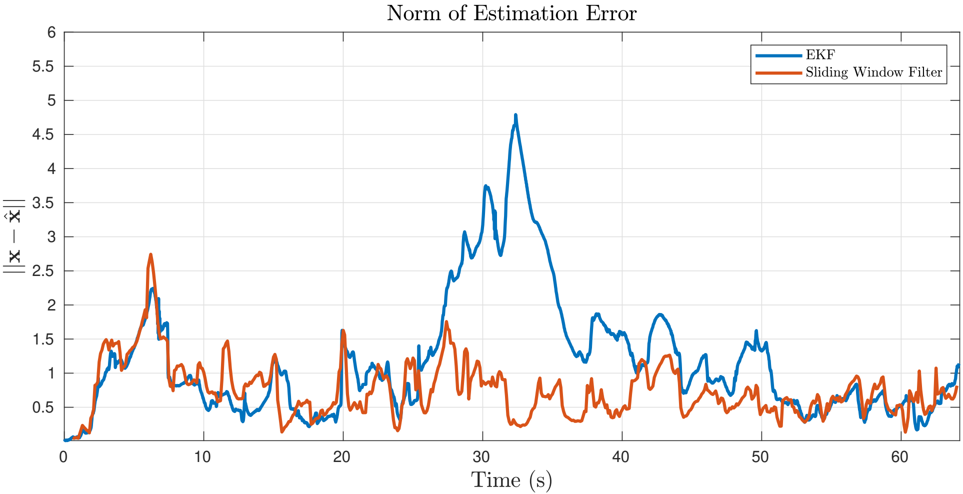


Figure 1: Estimation performance using real data of a quadrotor with an IMU and distance measurements to a landmark.

Information Form of the Sliding Window Filter

- Obtaining $\mathbf{\Sigma}_m$ requires us to invert $(\mathbf{H}_m^\mathsf{T}\mathbf{W}_m\mathbf{H}_m)$, which can be expensive for large window sizes.
- We can completely avoid doing this if we instead parameterize

$$p(\mathbf{x}_m|\check{\mathbf{x}}_0, \mathbf{u}_{0:m-1}, \mathbf{y}_{0:m-1}) = \mathcal{N}^{-1}(\bar{\boldsymbol{\eta}}, \bar{\mathbf{\Lambda}}) \tag{18}$$

  using the information form.
- We have easy access to the following matrices, which we can "split up" into sub-blocks as follows

$$\mathbf{H}_m^\mathsf{T}\mathbf{W}_m\mathbf{H}_m = \mathbf{\Lambda}_{0:m} \triangleq \begin{bmatrix} \mathbf{\Lambda}_{0:m-1} & \mathbf{\Lambda}_{0:m-1,m} \\ \mathbf{\Lambda}_{m,0:m-1} & \mathbf{\Lambda}_m \end{bmatrix}$$
$$\mathbf{H}_m^\mathsf{T}\mathbf{W}_m\bar{\mathbf{e}}_m \triangleq \mathbf{b}_{0:m} \triangleq \begin{bmatrix} \mathbf{b}_{0:m-1} \\ \mathbf{b}_m \end{bmatrix}$$

- **The goal is to find expressions for $\bar{\eta}, \bar{\Lambda}$, as a function of the blocks of $\Lambda_{0:m}, \mathbf{b}_{0:m}$.**

Information Form of the Sliding Window Filter

- Recall that the mean and covariance of a Gaussian approximation to $p(\mathbf{x}_{0:m}|\check{\mathbf{x}}_0, \mathbf{u}_{0:m-1}, \mathbf{y}_{0:m-1})$ are given by

$$\boldsymbol{\mu}_{0:m} = \begin{bmatrix} \boldsymbol{\mu}_{0:m-1} \\ \boldsymbol{\mu}_m \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{x}}_{0:m-1} \\ \hat{\mathbf{x}}_m \end{bmatrix} - (\mathbf{H}_m^\mathsf{T} \mathbf{W}_m \mathbf{H}_m)^{-1} \mathbf{H}_m^\mathsf{T} \mathbf{W}_m \bar{\mathbf{e}}_m, \tag{19}$$

$$\boldsymbol{\Sigma}_{0:m} = \begin{bmatrix} \boldsymbol{\Sigma}_{0:m-1} & \boldsymbol{\Sigma}_{0:m-1,m} \\ \boldsymbol{\Sigma}_{m,0:m-1} & \boldsymbol{\Sigma}_m \end{bmatrix} = (\mathbf{H}_m^\mathsf{T} \mathbf{W}_m \mathbf{H}_m)^{-1}, \tag{20}$$

where

$$\bar{\mathbf{e}}_m = \mathbf{e}_m(\hat{\mathbf{x}}_{0:m}), \qquad \mathbf{H}_m = \left. \frac{\partial \mathbf{e}_m(\mathbf{x})}{\partial \mathbf{x}} \right|_{\hat{\mathbf{x}}_{0:m}}. \tag{21}$$

- It follows that the information matrix and information vector are

$$\boldsymbol{\Lambda}_{0:m} \triangleq \begin{bmatrix} \boldsymbol{\Lambda}_{0:m-1} & \boldsymbol{\Lambda}_{0:m-1,m} \\ \boldsymbol{\Lambda}_{m,0:m-1} & \boldsymbol{\Lambda}_m \end{bmatrix} = \boldsymbol{\Sigma}_{0:m}^{-1} = \mathbf{H}_m^\mathsf{T} \mathbf{W}_m \mathbf{H}_m \tag{22}$$

$$\boldsymbol{\eta}_{0:m} \triangleq \begin{bmatrix} \boldsymbol{\eta}_{0:m-1} \\ \boldsymbol{\eta}_m \end{bmatrix} = \boldsymbol{\Lambda}_{0:m} \boldsymbol{\mu}_{0:m} = \boldsymbol{\Lambda}_{0:m} \begin{bmatrix} \hat{\mathbf{x}}_{0:m-1} \\ \hat{\mathbf{x}}_m \end{bmatrix} - \mathbf{H}_m^\mathsf{T} \mathbf{W}_m \bar{\mathbf{e}}_m \tag{23}$$

- We seek to find $\bar{\boldsymbol{\Lambda}} = \boldsymbol{\Sigma}_m^{-1}$ and $\bar{\boldsymbol{\eta}} = \bar{\boldsymbol{\Lambda}} \boldsymbol{\mu}_m$.

Recall Marginalization Theorems

Theorem (Marginalization)

Given the joint Gaussian probability density function

$$p(\mathbf{x}, \mathbf{y}) = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu}_x \\ \boldsymbol{\mu}_y \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}_{xx} & \boldsymbol{\Sigma}_{xy} \\ \boldsymbol{\Sigma}_{yx} & \boldsymbol{\Sigma}_{yy} \end{bmatrix} \right) = \mathcal{N}^{-1}\left(\begin{bmatrix} \boldsymbol{\eta}_x \\ \boldsymbol{\eta}_y \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Lambda}_{xx} & \boldsymbol{\Lambda}_{xy} \\ \boldsymbol{\Lambda}_{yx} & \boldsymbol{\Lambda}_{yy} \end{bmatrix} \right),$$

the marginal pdf $p(\mathbf{x}) = \int_{-\infty}^{\infty} p(\mathbf{x}, \mathbf{y}) \mathrm{d}\mathbf{y}$ is given in covariance form as

$$p(\mathbf{x}) = \mathcal{N}(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_{xx}), \tag{24}$$

or in information form as

$$p(\mathbf{x}) = \mathcal{N}^{-1}(\bar{\boldsymbol{\eta}}, \bar{\boldsymbol{\Lambda}}), \tag{25}$$

where

$$\bar{\boldsymbol{\eta}} = \boldsymbol{\eta}_x - \boldsymbol{\Lambda}_{xy} \boldsymbol{\Lambda}_{yy}^{-1} \boldsymbol{\eta}_y \qquad , \qquad \bar{\boldsymbol{\Lambda}} = \boldsymbol{\Lambda}_{xx} - \boldsymbol{\Lambda}_{xy} \boldsymbol{\Lambda}_{yy}^{-1} \boldsymbol{\Lambda}_{yx}. \tag{26}$$

Information Form - Getting $\bar{\mathbf{\Lambda}}$

- The marginalization theorem allows us to directly obtain $\bar{\mathbf{\Lambda}}$ with

$$\bar{\mathbf{\Lambda}} = \mathbf{\Lambda}_m - \mathbf{\Lambda}_{m,0:m-1}\mathbf{\Lambda}_{0:m-1}^{-1}\mathbf{\Lambda}_{0:m-1,m}. \tag{27}$$

- Similarly, we can also use the marginalization theorem to obtain $\bar{\boldsymbol{\eta}}$, but this requires a bit more algebra...

Information Form - Getting $\bar{\boldsymbol{\eta}}$

- From the marginalization theorem,

$$\bar{\boldsymbol{\eta}} = \boldsymbol{\eta}_m - \boldsymbol{\Lambda}_{m,0:m-1}\boldsymbol{\Lambda}_{0:m-1}^{-1}\boldsymbol{\eta}_{0:m-1}, \tag{28}$$

where we have

$$\boldsymbol{\eta}_{0:m} = \boldsymbol{\Lambda}_{0:m}\boldsymbol{\mu}_{0:m} \tag{29}$$

$$\begin{bmatrix} \boldsymbol{\eta}_{0:m-1} \\ \boldsymbol{\eta}_m \end{bmatrix} = \begin{bmatrix} \boldsymbol{\Lambda}_{0:m-1} & \boldsymbol{\Lambda}_{0:m-1,m} \\ \boldsymbol{\Lambda}_{m,0:m-1} & \boldsymbol{\Lambda}_m \end{bmatrix} \begin{bmatrix} \hat{\mathbf{x}}_{0:m-1} \\ \hat{\mathbf{x}}_m \end{bmatrix} - \underbrace{\mathbf{H}_m^\mathsf{T}\mathbf{W}_m\bar{\mathbf{e}}_m}_{\triangleq \begin{bmatrix} \mathbf{b}_{0:m-1} \\ \mathbf{b}_m \end{bmatrix}} . \tag{30}$$

- Substituting in expressions for $\boldsymbol{\eta}_{0:m-1}$ and $\boldsymbol{\eta}_m$ from (30) into (28), and doing the algebra eventually yields

$$\bar{\boldsymbol{\eta}} = \bar{\boldsymbol{\Lambda}}\hat{\mathbf{x}}_m - \left(\mathbf{b}_m - \boldsymbol{\Lambda}_{m,0:m-1}\boldsymbol{\Lambda}_{0:m-1}^{-1}\mathbf{b}_{0:m-1}\right) \tag{31}$$

Information Form - Prior Distribution

- Now that we have obtained expressions for $\bar{\boldsymbol{\eta}}, \bar{\boldsymbol{\Lambda}}$, the new prior distribution can be expressed in information form as

$$\begin{aligned} p(\mathbf{x}_m|\check{\mathbf{x}}_0, \mathbf{u}, \mathbf{y}_{0:m-1}) &= \mathcal{N}^{-1}(\bar{\boldsymbol{\eta}}, \bar{\boldsymbol{\Lambda}}), && (32) \\ &= \beta \exp\left(-\frac{1}{2}\mathbf{x}_m^\mathsf{T}\bar{\boldsymbol{\Lambda}}\mathbf{x}_m + \bar{\boldsymbol{\eta}}^\mathsf{T}\mathbf{x}_m\right), && (33) \\ &= \kappa \exp\left(-\frac{1}{2}(\mathbf{x}_m - \hat{\mathbf{x}}_m)^\mathsf{T}\bar{\boldsymbol{\Lambda}}(\mathbf{x}_m - \hat{\mathbf{x}}_m)\right. && (34) \\ &\qquad \left. -(\mathbf{b}_m - \boldsymbol{\Lambda}_{m,0:m-1}\boldsymbol{\Lambda}_{0:m-1}^{-1}\mathbf{b}_{0:m-1})^\mathsf{T}\mathbf{x}_m\right), \end{aligned}$$

where β and κ are normalization constants.

A few algebra steps were skipped going from (33) to (34).

Optimization Problem in Information Form

In information form, the least-squares problem gains an additional linear term

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \left(\frac{1}{2} \mathbf{e}(\mathbf{x})^{\mathsf{T}} \mathbf{W} \mathbf{e}(\mathbf{x}) + (\mathbf{b}_m - \mathbf{\Lambda}_{m,0:m-1} \mathbf{\Lambda}_{0:m-1}^{-1} \mathbf{b}_{0:m-1})^{\mathsf{T}} \mathbf{x}_m \right) \tag{35}$$

where

$$\mathbf{e}(\mathbf{x}) = \begin{bmatrix} \mathbf{x}_m - \hat{\mathbf{x}}_m \\ \mathbf{x}_{m+1} - \mathbf{f}(\mathbf{x}_m, \mathbf{u}_m, \mathbf{0}) \\ \vdots \\ \mathbf{x}_{k_2} - \mathbf{f}(\mathbf{x}_{k_2-1}, \mathbf{u}_{k_2-1}, \mathbf{0}) \\ \mathbf{y}_m - \mathbf{g}(\mathbf{x}_m, \mathbf{0}) \\ \vdots \\ \mathbf{y}_{k_2} - \mathbf{g}(\mathbf{x}_{k_2}, \mathbf{0}) \end{bmatrix}, \tag{36}$$

$$\mathbf{W} = \operatorname{diag}(\bar{\mathbf{\Lambda}}, \mathbf{Q}_{m+1}^{-1}, \ldots, \mathbf{Q}_{k_2}^{-1}, \mathbf{R}_m^{-1}, \ldots, \mathbf{R}_{k_2}^{-1}). \tag{37}$$

The Gauss-Newton algorithm can still be used with this additional linear term.

References

These slides are based on [1] [2] [3] [4]

[1] T. Barfoot, *State Estimation for Robotics*. Toronto, ON: Cambridge University Press, 2019.

[2] T. C. Dong-Si and A. I. Mourikis, “Motion tracking with fixed-lag smoothing: Algorithm and consistency analysis,” *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 5655–5662, 2011.

[3] G. Sibley, “A Sliding Window Filter for SLAM,” University of Southern California, Tech. Rep., 2006.

[4] R. M. Eustice, H. Singh, and J. J. Leonard, “Exactly sparse delayed-state filters for view-based SLAM,” *IEEE Transactions on Robotics*, vol. 22, no. 6, pp. 1100–1114, 2006.