ECE276A: Sensing & Estimation in Robotics
Lecture 15: Kalman Smoothing and Factor Graphs

Lecturer:

Nikolay Atanasov: natanasov@ucsd.edu

Teaching Assistants:

Siwei Guo: s9guo@eng.ucsd.edu

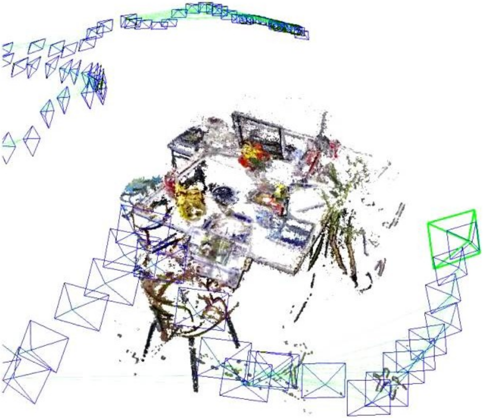
Anwesan Pal: a2pal@eng.ucsd.edu

UC San Diego

JACOBS SCHOOL OF ENGINEERING

Electrical and Computer Engineering

Visual-Inertial SLAM



Simultaneous Localization and Mapping

- **Goal**: estimate the robot pose while creating a map of the environment
- Known as **Structure from Motion** (SfM) in computer vision.
- In SfM usually the only available sensor is a camera, while in SLAM the robot might have odometry (e.g., IMU) and other sensors (e.g., lidar)
- **2-D Landmark-based SLAM**: maintains a robot state $x_t := (p_t, \phi_t) \in SE(2)$ and n landmark states $l_{t,i} \in \mathbb{R}^2$, $i = 1, \ldots, n$:

$$s_t := \begin{pmatrix} p_t \\ \phi_t \\ l_{t,1} \\ \vdots \\ l_{t,n} \end{pmatrix} \in \mathbb{R}^{3+2n} \qquad s_t \sim \mathcal{N}\left(\begin{pmatrix} \mu^x_{t|t} \\ \mu^l_{t|t} \end{pmatrix}, \begin{bmatrix} \Sigma^x_{t|t} & \Sigma^{xl}_{t|t} \\ \Sigma^{lx}_{t|t} & \Sigma^l_{t|t} \end{bmatrix} \right)$$

2-D Landmark-based SLAM Prediction

- **Motion model**: differential drive robot and static landmarks:

$$\begin{pmatrix} p_{t+1} \\ \phi_{t+1} \end{pmatrix} = a\left(\begin{pmatrix} p_t \\ \phi_t \end{pmatrix}, \begin{pmatrix} v_t \\ \omega_t \end{pmatrix} + w_t\right), \qquad w_t \sim \mathcal{N}(0, W)$$

$$l_{t+1,i} = l_{t,i}$$

$$a\left(\begin{pmatrix} p_t \\ \phi_t \end{pmatrix}, \begin{pmatrix} v_t \\ \omega_t \end{pmatrix}\right) := \begin{pmatrix} p_t \\ \phi_t \end{pmatrix} + \tau \begin{pmatrix} v_t \textbf{sinc}\left(\frac{\omega_t \tau}{2}\right) \cos\left(\phi_t + \frac{\omega_t \tau}{2}\right) \\ v_t \textbf{sinc}\left(\frac{\omega_t \tau}{2}\right) \sin\left(\phi_t + \frac{\omega_t \tau}{2}\right) \\ \omega_t \end{pmatrix}$$

- **Motion Model Jacobian** (see Lecture 9): $A_t := \frac{da}{dx}(\mu^x_{t|t}, u_t) \in \mathbb{R}^{3\times 3}$ and $Q_t := \frac{da}{dw}(\mu^x_{t|t}, u_t) \in \mathbb{R}^{3\times 2}$

- **Prediction**

$$\begin{bmatrix} \Sigma^x_{t+1|t} & \Sigma^{xl}_{t+1|t} \\ \Sigma^{lx}_{t+1|t} & \Sigma^{l}_{t+1|t} \end{bmatrix} = \begin{bmatrix} A_t & 0 \\ 0 & I_{2n\times 2n} \end{bmatrix} \begin{bmatrix} \Sigma^x_{t|t} & \Sigma^{xl}_{t|t} \\ \Sigma^{lx}_{t|t} & \Sigma^{l}_{t|t} \end{bmatrix} \begin{bmatrix} A_t^T & 0 \\ 0 & I_{2n\times 2n} \end{bmatrix} + \begin{bmatrix} Q_t \\ \mathbf{0}_{2n\times 2} \end{bmatrix} W \begin{bmatrix} Q_t^T & \mathbf{0}_{2\times 2n} \end{bmatrix}$$

2-D Landmark-based SLAM Prediction

- The upper left 3×3 element of the covariance is computed in the same way as if the robot were propagating its covariance without considering the presence of landmarks
- The correlation $\Sigma_{t|t}^{xl} \in \mathbb{R}^{3\times 2n}$ between the robot pose and the landmarks is multiplied by A_t and hence changes over time proportionally to the orientation uncertainty of the robot (see the differential-drive Jacobian)
- The landmark covariances $\Sigma_{t|t}^{l} \in \mathbb{R}^{2n\times 2n}$ do not change during propagation because of the static landmark assumption
- In practice there are many propagation steps in between update steps. The propagation can be simplified by only propagating the robot covariance $\Sigma_{t|t}^{x}$ and keeping track of $A_{t+k} \dots A_{t+1} A_t$

2-D Landmark-based SLAM Update

- **Observation model**: relative position measurements:

$$z_{t,i} = h(x_t, l_{t,i}) + v_{t,i} := R^T(\phi_t)(l_{t,i} - p_t) + v_{t,i}, \qquad v_{t,i} \sim \mathcal{N}(0, V_i)$$

- **Observation model Jacobian** (see Lecture 9):

$$\frac{dh}{dp} = -R^T(\phi) \qquad \frac{dh}{d\phi} = R^T(\phi) J^T (l_i - p) \qquad \frac{dh}{dl_i} = R^T(\phi)$$

$$H_{t,i} := [-R^T(\mu^{\phi}_{t|t}),\ R^T(\mu^{\phi}_{t|t}) J^T(\mu^{l_i}_{t|t} - \mu^{p}_{t|t}),\ 0,\ \cdots,\ 0,\ R^T(\mu^{\phi}_{t|t}),\ 0,\ \cdots,\ 0] \in \mathbb{R}^{2\times(3+2n)}$$

$$H_t := \begin{bmatrix} H_{t,1} \\ \vdots \\ H_{t,n} \end{bmatrix} \in \mathbb{R}^{(2n)\times(3+2n)} \qquad V := \begin{bmatrix} V_1 & & \\ & \ddots & \\ & & V_n \end{bmatrix}$$

- **Update**: due to the independence assumptions there is a special structure but it is hard to see in the covariance update

$$\begin{bmatrix} \Sigma^{x}_{t+1|t+1} & \Sigma^{xl}_{t+1|t+1} \\ \Sigma^{lx}_{t+1|t+1} & \Sigma^{l}_{t+1|t+1} \end{bmatrix} = (I - K_{t+1|t} H_{t+1}) \begin{bmatrix} \Sigma^{x}_{t+1|t} & \Sigma^{xl}_{t+1|t} \\ \Sigma^{lx}_{t+1|t} & \Sigma^{l}_{t+1|t} \end{bmatrix}$$

$$K_{t+1|t} = \Sigma_{t+1|t} H^T_{t+1} \left(H_{t+1} \Sigma_{t+1|t} H^T_{t+1} + V \right)^{-1}$$

Information Filter

- Uses the natural Gaussian parameterization $x \sim \mathcal{G}(\nu, \Omega)$ where $\nu = \Sigma^{-1}\mu$ and $\Omega = \Sigma^{-1}$
- Uses the matrix inversion lemma to convert the Kalman filter covariance equations to their information matrix counterparts

Prior: $x_t \mid z_{0:t}, u_{0:t-1} \sim \mathcal{G}(\nu_{t|t}, \Omega_{t|t})$

Motion model: $x_{t+1} = Ax_t + Bu_t + w_t, \quad w_t \sim \mathcal{G}(0, W^{-1})$

Observation model: $z_t = Hx_t + v_t, \quad v_t \sim \mathcal{G}(0, V^{-1})$

Prediction:
$$\begin{aligned}\nu_{t+1|t} &= (I - C_{t|t})A^{-T}\nu_{t|t}\\ \Omega_{t+1|t} &= (I - C_{t|t})A^{-T}\Omega_{t|t}A^{-1}(I - C_{t|t}^T) + C_{t|t}W^{-1}C_{t|t}^T\end{aligned}$$

Information Gain: $C_{t|t} = A^{-T}\Omega_{t|t}A^{-1}\left(A^{-T}\Omega_{t|t}A^{-1} + W^{-1}\right)^{-1}$

Update:
$$\begin{aligned}\nu_{t+1|t+1} &= \nu_{t+1|t} + H^T V^{-1} z_{t+1}\\ \Omega_{t+1|t+1} &= \Omega_{t+1|t} + H^T V^{-1} H\end{aligned}$$

2-D Landmark-based SLAM Update in Information Space

- **Observation model**: $z_{t,i} = R^T(\phi_t)(l_{t,i} - p_t) + v_{t,i}$, $v_{t,i} \sim \mathcal{N}(0, V_i)$
- **Observation model Jacobian**:

$$H_{t,i} := [-R^T(\mu^{\phi}_{t|t}),\ R^T(\mu^{\phi}_{t|t})J^T(\mu^{l_i}_{t|t} - \mu^{p}_{t|t}),\ 0,\ \cdots,\ 0,\ R^T(\mu^{\phi}_{t|t}),\ 0,\ \cdots,\ 0] \in \mathbb{R}^{2\times(3+2n)}$$

$$H_t := \begin{bmatrix} H_{t,1} \\ \vdots \\ H_{t,n} \end{bmatrix} \in \mathbb{R}^{(2n)\times(3+2n)} \qquad V := \begin{bmatrix} V_1 & & \\ & \ddots & \\ & & V_n \end{bmatrix}$$

- **Information space update**: the information from individual measurements is added sequentially to the information matrix

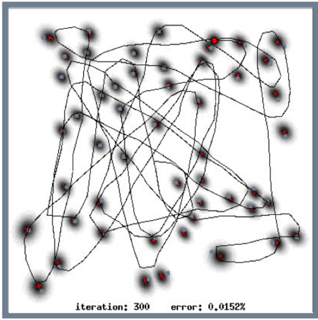
$$\nu_{t+1|t+1} = \nu_{t+1|t} + H_{t+1}^T V^{-1} z_{t+1} = \nu_{t+1|t} + \sum_{i=1}^{n} H_{t+1,i} V_i^{-1} z_{t+1,i}$$

$$\Omega_{t+1|t+1} = \Omega_{t+1|t} + H_{t+1}^T V^{-1} H_{t+1} = \Omega_{t+1|t} + \sum_{i=1}^{n} H_{t+1,i} V_i^{-1} H_{t+1,i}^T$$

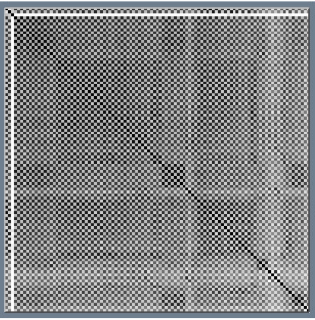
- Since for each measurement $z_{t,i}$, only the corresponding block in the information matrix is updated, $\Omega_{t|t}$ remains **sparse** over time

Information Matrix Sparsity

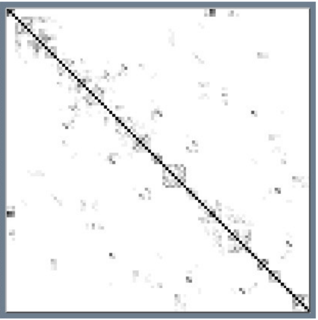
- Ω_{ij} tells us the correlation strength among landmarks and the robot pose
- Most landmarks have only a small number of strong correlations
- The Info matrix can be interpreted as a graph of constraints (edges) between variables (nodes). Missing edges indicate conditional independence.
- **Sparsification**: remove weak correlations to improve efficiency&memory



Pose-map distribution



Covariance matrix

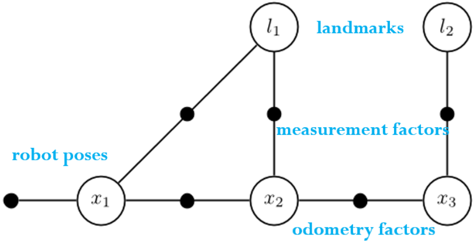


Information matrix

EKF vs Sparse Extended Info Filter SLAM

- KF: efficient prediction, slow correction
- IF: slow prediction, efficient correction
- EKF SLAM Complexity:
  - **Time complexity**: cubic in the measurement dimension but dominated by the number of landmarks: $O(n^2)$
  - **Memory complexity**: $O(n^2)$
  - EKF SLAM is computationally intractable for large maps
- SEIF SLAM Complexity:
  - Neglects correlations via sparsification and only approximates the mean (since computing $\mu = \Omega^{-1}\nu$ is very costly)
  - **Time complexity**: roughly constant
  - **Memory complexity**: $O(n)$
  - Inferior quality compared to EKF SLAM due to sparsification and approximate mean recovery
- Further reading:
  - **EKF SLAM**: Thrun et al., "Probabilistic Robotics," Ch. 10
  - **SEIF SLAM**: Thrun et al., "Probabilistic Robotics," Ch. 12

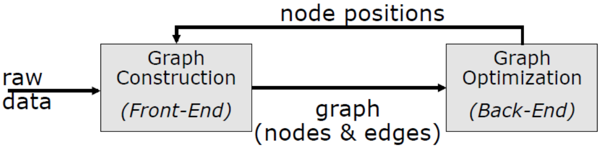
Factor Graph



- A graphical model capturing the first-order Markov assumptions

- **Front-end**: constructs the graph using dense scan-matching, feature matching or descriptor matching
  1. **Nodes**: variables to be estimated (e.g., robot and landmark $SE(3)$ poses)
  2. **Edges** (called **factors**): have associated measurement error functions and information matrices, defining a Mahalonobis norm on the error
     - **Odometry**: $e_{ij}(x_i, x_j) = (x_j \ominus x_i) \ominus z_{ij}$ and $\Omega_{ij} = W^{-1}$
     - **Camera**: $e_{ij}(x_i, x_j) = z_{ij} - h(x_i, x_j)$ and $\Omega_{ij} = V^{-1}$

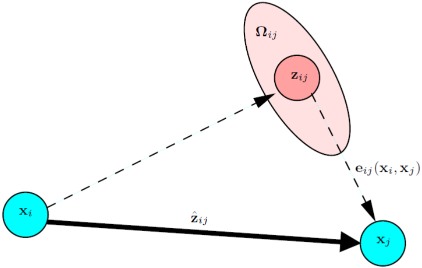
- **Back-end**: performs inference over the graph



Inference over Factor Graphs

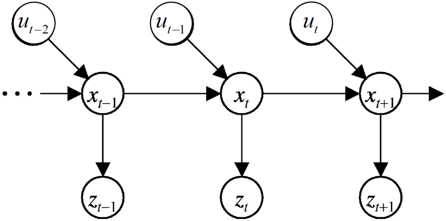
- Inference over the graph: a nonlinear least-squares problem:

$$\arg\max_{x} \sum_{(i,j)\in E} \underbrace{e_{ij}(x)^T \Omega_{ij} e_{ij}(x)}_{F_{ij}(x)}$$



- Linearization of the factors $F_{ij}(x)$ leads to a **sparse linear system**
- Assumptions:
  - A "good" initial guess is available
  - The error functions are smooth in the neighborhood of the minima
- Iterative linearization:
  1. linearize the error functions $e_{ij}(x)$ around the current guess
  2. compute the gradient of the quadratic objective $\sum_{(i,j)\in E} F_{ij}(x)$, set it equal to zero, and solve the resulting linear system
  3. update the current guess and repeat
- The linearization points can be corrected iteratively via the **Gauss-Newton** or **Levenberg-Marquardt** algorithms

Bayes Filter



- **Motion model**:
  $x_{t+1} = a(x_t, u_t, w_t) \sim p_a(\cdot \mid x_t, u_t)$
- **Observation model**:
  $z_t = h(x_t, v_t) \sim p_h(\cdot \mid x_t)$
- **Filtering**: keeps track of
$$p_{t|t}(x_t) := p(x_t \mid z_{0:t}, u_{0:t-1})$$
$$p_{t+1|t}(x_{t+1}) := p(x_{t+1} \mid z_{0:t}, u_{0:t})$$
- **Bayes filter**:
$$p_{t+1|t+1}(x_{t+1}) = \underbrace{\overbrace{\frac{1}{p(z_{t+1}|z_{0:t}, u_{0:t})}}^{\frac{1}{\eta_{t+1}}} p_h(z_{t+1} \mid x_{t+1}) \overbrace{\int p_a(x_{t+1} \mid x_t, u_t) p_{t|t}(x_t) dx_t}^{\textbf{Predict: } p_{t+1|t}(x_{t+1})}}_{\textbf{Update}}$$
- **Joint distribution**:
$$p(x_{0:T}, z_{0:T}, u_{0:T-1}) = \underbrace{p_{0|0}(x_0)}_{\text{prior}} \prod_{t=0}^{T} \underbrace{p_h(z_t \mid x_t)}_{\text{observation model}} \prod_{t=0}^{T} \underbrace{p_a(x_t \mid x_{t-1}, u_{t-1})}_{\text{motion model}}$$

Bayesian Smoothing

- **Smoothing**: keeps track of
$$p_{t|t}(x_{0:t}) := p(x_{0:t} \mid z_{0:t}, u_{0:t-1})$$
$$p_{t+1|t}(x_{0:t+1}) := p(x_{0:t+1} \mid z_{0:t}, u_{0:t})$$

- Forward pass (**Bayes filter**): compute $p(x_{t+1} \mid z_{0:t+1}, u_{0:t})$ and $p(x_{t+1} \mid z_{0:t}, u_{0:t})$ for $t = 0, \ldots, T$

- Backward pass (**Bayes smoother**): for $t = T-1, \ldots, 0$ compute:

$$p(x_t \mid z_{0:T}, u_{0:T-1}) \xlongequal[\text{Probability}]{\text{Total}} \int p(x_t \mid x_{t+1}, z_{0:T}, u_{0:T-1}) p(x_{t+1} \mid z_{0:T}, u_{0:T-1}) dx_{t+1}$$

$$\xlongequal[\text{Assumption}]{\text{Markov}} \int p(x_t \mid x_{t+1}, z_{0:t}, u_{0:t}) p(x_{t+1} \mid z_{0:T}, u_{0:T-1}) dx_{t+1}$$

$$\xlongequal[\text{Rule}]{\text{Bayes}} \underbrace{p(x_t \mid z_{0:t}, u_{0:t-1})}_{\text{forward pass}} \int \left[\frac{\overbrace{p_a(x_{t+1} \mid x_t, u_t)}^{\text{motion model}} p(x_{t+1} \mid z_{0:T}, u_{0:T-1})}{\underbrace{p(x_{t+1} \mid z_{0:t}, u_{0:t})}_{\text{forward pass}}} \right] dx_{t+1}$$

Rauch-Tung-Striebel (Kalman) Smoothing

- Prior: $x_0 \sim \mathcal{N}(\mu_{0|0}, \Sigma_{0|0})$
- Motion model: $x_{t+1} = Ax_t + Bu_t + w_t$ with $w_t \sim \mathcal{N}(0, W)$
- Observation model: $z_t = Hx_t + v_t$ with $v_t \sim \mathcal{N}(0, V)$
- Forward pass (**Kalman filter**): compute $\left\{\left(\mu_{t|t}, \Sigma_{t|t}\right)\right\}_{t=1}^{T}$ and $\left\{\left(\mu_{t+1|t}, \Sigma_{t+1|t}\right)\right\}_{t=0}^{T-1}$
- Backward pass (**Kalman smoother**): let $\left(\mu^S_{T|T}, \Sigma^S_{T|T}\right) := \left(\mu_{T|T}, \Sigma_{T|T}\right)$ and compute the smoothed estimates $\left\{\left(\mu^S_{t|t}, \Sigma^S_{t|t}\right)\right\}_{t=T-1}^{0}$ as follows:

$$
\begin{aligned}
&\text{for } \ t = T-1, \ldots, 0 \\
&\qquad G_t = \Sigma_{t|t} A^T \left(\Sigma_{t+1|t}\right)^{-1} \\
&\qquad \mu^S_{t|t} = \mu_{t|t} + G_t \left(\mu^S_{t+1|t+1} - \mu_{t+1|t}\right) \\
&\qquad \Sigma^S_{t|t} = \Sigma_{t|t} + G_t \left(\Sigma^S_{t+1|t+1} - \Sigma_{t+1|t}\right) G_t^T
\end{aligned}
$$

Extended Kalman Smoothing via Least Squares

- **Prior**: $x_0 \sim \mathcal{N}(\mu_0, \Sigma_{0|0})$
- **Noise**: $w_t \sim \mathcal{N}(0, W)$ and $v_t \sim \mathcal{N}(0, V)$
- **Linearization point**: initial estimate $\mu_{0:T}$, e.g., from odometry
- **Motion model linearization**:
  $x_{t+1} = a(x_t, u_t, w_t) \approx a(\mu_t, u_t, 0) + A_t(x_t - \mu_t) + Q_t w_t$
- **Observation model linearization**:
  $z_t = h(x_t, v_t) \approx h(\mu_t, 0) + H_t(x_t - \mu_t) + R_t v_t$
- **Jacobians**: $A_t := \frac{da}{dx}(\mu_t, u_t, 0)$ and $Q_t := \frac{da}{dw}(\mu_t, u_t, 0)$ and $H_t := \frac{dh}{dx}(\mu_t, 0)$ and $R_t := \frac{dh}{dv}(\mu_t, 0)$
- **Error model**: $e_t := x_t - \mu_t$ and $\eta_{t+1} := \mu_{t+1} - a(\mu_t, u_t, 0)$ and $\zeta_t := z_t - h(\mu_t, 0)$

$$
\begin{aligned}
e_{t+1} + \eta_{t+1} &= A_t e_t + w'_t, \qquad & w'_t &\sim \mathcal{N}(0, \overbrace{Q_t W Q_t^T}^{W_t}) \\
\zeta_{t+1} &= H_{t+1} e_{t+1} + v'_{t+1}, \qquad & v'_{t+1} &\sim \mathcal{N}(0, \underbrace{R_{t+1} V R_{t+1}^T}_{V_{t+1}})
\end{aligned}
$$

Extended Kalman Smoothing via Least Squares

- **Joint distribution**:

$$p(x_{0:T}, z_{0:T}, u_{0:T-1}) = \underbrace{p_{0|0}(x_0)}_{\text{prior}} \prod_{t=0}^{T} \underbrace{p_h(z_t \mid x_t)}_{\text{observation model}} \prod_{t=1}^{T} \underbrace{p_a(x_t \mid x_{t-1}, u_{t-1})}_{\text{motion model}}$$

- SLAM via MLE leads to nonlinear least squares:

$$\arg\max_{x_{0:T}} \log p(x_{0:T}, z_{0:T}, u_{0:T-1}) \xlongequal[\text{initial guess } \mu_{0:T}]{\text{linearize around}}$$

$$\approx \mu_{0:T} + \arg\min_{e_{0:T}} \left\{ \|e_0\|^2_{\Sigma_{0|0}} + \sum_{t=0}^{T} \|\zeta_t - H_t e_t\|^2_{V_t} + \sum_{t=1}^{T} \|\eta_t + e_t - A_t e_{t-1}\|^2_{W_t} \right\} \xlongequal[\|x\|_\Sigma := \sqrt{x^T \Sigma^{-1} x} = \|\Sigma^{-1/2} x\|_2]{\text{Mahalonobis Distance}}$$

$$= \mu_{0:T} + \arg\min_{e_{0:T}} \left\{ \|\Sigma_{0|0}^{-1/2} e_0\|_2^2 + \sum_{t=0}^{T} \| V_t^{-1/2} (\zeta_t - H_t e_t) \|_2^2 + \sum_{t=1}^{T} \| W_{t-1}^{-1/2} (\eta_t + e_t - A_t e_{t-1}) \|_2^2 \right\}$$

- Solve the linear least squares problem to obtain $e_{0:T}$
- Update the linearization points: $\mu'_{0:T} = \mu_{0:T} + e_{0:T}$
- Repeat by linearizing around $\mu'_{0:T}$

Sparse Least Squares

- $\left\| \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} - \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \right\|_2^2 = \|x_1 - y_1\|_2^2 + \|x_2 - y_2\|_2^2$ for $x_1, y_1 \in \mathbb{R}^{d_1}$, $x_2, y_2 \in \mathbb{R}^{d_2}$
- Using this we can write the least-squares problem in matrix notation:

$$\|\Sigma_{0|0}^{-1/2} e_0\|^2 + \sum_{t=0}^{T} \| V_t^{-1/2} (\zeta_t - H_t e_t) \|^2 + \sum_{t=1}^{T} \| W_{t-1}^{-1/2} (\eta_t + e_t - A_{t-1} e_{t-1}) \|^2$$

$$= \|\Sigma_{0|0}^{-1/2} e_0\|^2 + \left\| \begin{bmatrix} V_0^{-1/2} (\zeta_0 - H_0 e_0) \\ \vdots \\ V_T^{-1/2} (\zeta_T - H_T e_T) \end{bmatrix} \right\|_2^2 + \left\| \begin{bmatrix} W_0^{-1/2} (\eta_1 + e_1 - A_0 e_0) \\ \vdots \\ W_{T-1}^{-1/2} (\eta_T + e_T - A_{T-1} e_{T-1}) \end{bmatrix} \right\|_2^2$$

$$= \|\Sigma_{0|0}^{-1/2} e_0\|^2 + \left\| \begin{bmatrix} V_0^{-1/2} H_0 & & \\ & \ddots & \\ & & V_T^{-1/2} H_T \end{bmatrix} \begin{pmatrix} e_0 \\ \vdots \\ e_T \end{pmatrix} - \begin{bmatrix} V_0^{-1/2} \zeta_0 \\ \vdots \\ V_T^{-1/2} \zeta_T \end{bmatrix} \right\|_2^2$$

$$+ \left\| \begin{bmatrix} W_0^{-1/2} A_0 & -W_0^{-1/2} & & \\ & W_1^{-1/2} A_1 & \ddots & \\ & & \ddots & -W_{T-1}^{-1/2} \\ & & & W_{T-1}^{-1/2} A_{T-1} \end{bmatrix} \begin{pmatrix} e_0 \\ \vdots \\ e_T \end{pmatrix} - \begin{bmatrix} W_0^{-1/2} \eta_1 \\ \vdots \\ W_{T-1}^{-1/2} \eta_T \end{bmatrix} \right\|_2^2$$

Sparse Least Squares

$$
\left\| \underbrace{\begin{bmatrix}
\Sigma_{0|0}^{-1/2} & & & \\
V_0^{-1/2} H_0 & & & \\
& \ddots & & \\
& & \ddots & \\
& & & V_T^{-1/2} H_T \\
W_0^{-1/2} A_0 & -W_0^{-1/2} & & \\
& W_1^{-1/2} A_1 & \ddots & \\
& & \ddots & -W_{T-1}^{-1/2} \\
& & & W_{T-1}^{-1/2} A_{T-1}
\end{bmatrix}}_{J}
\begin{pmatrix} e_0 \\ \vdots \\ e_T \end{pmatrix}
-
\underbrace{\begin{bmatrix}
0 \\
V_0^{-1/2} \zeta_0 \\
\vdots \\
\vdots \\
V_T^{-1/2} \zeta_T \\
W_0^{-1/2} \eta_1 \\
W_1^{-1/2} \eta_2 \\
\vdots \\
W_{T-1}^{-1/2} \eta_T
\end{bmatrix}}_{b}
\right\|_2^2
$$

$$= \| J e_{0:T} - b \|_2^2$$

Sparse Least Squares

- Via linearization, we managed to reduce the SLAM problem to:

$$\arg\max_{x_{0:T}} \log p(x_{0:T}, z_{0:T}, u_{0:T-1}) \xrightarrow[\text{initial guess } \mu_{0:T}]{\text{linearize around}} \mu_{0:T} + \arg\min_{e_{0:T}} \|Je_{0:T} - b\|_2^2$$

- The matrix of Jacobians J is **sparse**
- $J^T J$ is the **information matrix** of the joint Gaussian distribution of $x_{0:T} \mid z_{0:T}, u_{0:T-1}$
- Setting the gradient to zero leads to the **Normal equations**:

$$\boxed{J^T J e_{0:T} = J^T b}$$

- Can be solved via **Cholesky decomposition** of $J^T J$
- A more efficient and robust way, which avoids having to compute the information matrix $J^T J$ (which also squares the condition number), is **QR factorization**

Solution via QR Factorization

- QR factorization: $J = Q \begin{bmatrix} R \\ 0 \end{bmatrix} \in \mathbb{R}^{m \times n}$
- The number of variables (nodes) is n
- The number of constraints (factors) is m
- $R \in \mathbb{R}^{n \times n}$ is the **upper triangular square root information matrix** since $R^T R = J^T J$
- $Q \in \mathbb{R}^{m \times m}$ is an orthogonal matrix
- Solution via QR factorization:

$$
\begin{aligned}
\|J e_{0:T} - b\|_2^2 &= \left\| Q \begin{bmatrix} R \\ 0 \end{bmatrix} e_{0:T} - b \right\|_2^2 = \left\| Q^T Q \begin{bmatrix} R \\ 0 \end{bmatrix} e_{0:T} - Q^T b \right\|_2^2 \\
&= \left\| \begin{bmatrix} R \\ 0 \end{bmatrix} e_{0:T} - \begin{bmatrix} b'_1 \\ b'_2 \end{bmatrix} \right\|_2^2 = \|R e_{0:T} - b'_1\|_2^2 + \underbrace{\|b'_2\|_2^2}_{\text{residual}}
\end{aligned}
$$

- Since R is upper triangular, simple back-substitution can be used to compute $e^*_{0:T}$ — leading to a least squares estimate for the complete robot trajectory as well as all landmarks $x_{0:T}$ conditioned on all measurements $z_{0:T}$, $u_{0:T-1}$

Factor Graph SLAM Summary

- The factor graph view of SLAM leads to a nonlinear least squares problem
- Assuming an initial estimate of the robot trajectory and landmark poses is available (e.g., from odometry and triangulation of 2-D image features), we can use the Gauss-Newton algorithm to solve the nonlinear least squares problem
- Gauss-Newton iterates between linearizing the system and solving the resulting linear equation to update the pose-landmark estimates
- Assuming a Gaussian distribution for the constraints is not always the best choice in the presence of outliers. A heavy-tailed distribution can be used for outlier rejection as in Lecture 14.
- **Loop closure**: observing previously seen landmarks generates constraints between non-successive robot poses