TTK4250

Week 12

Graphical Models: Towards Graph-based SLAM

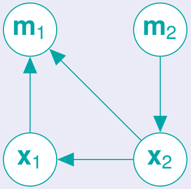
Edmund Førland Brekke

7.November 2024

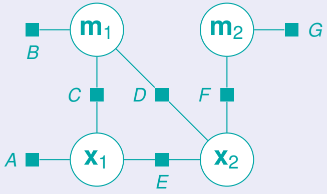
Outline

Recap from last week

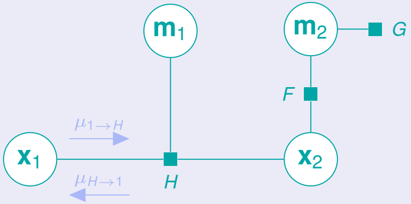
Bayesian networks



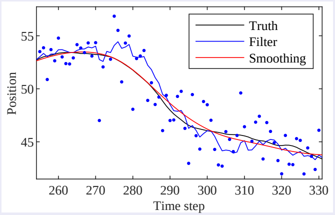
Factor graphs



Belief propagation



Smoothing



Outline

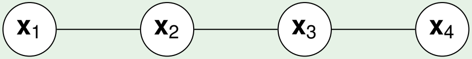
Markov random fields

Pairwise Markov random fields

A pairwise MRF provides a factorization of the pdf of the form

$$p(x_1, \ldots, x_N) = \frac{1}{Z} \prod_i \psi(x_i) \prod_{ij} \psi(x_i, x_j)$$

Example: The dynamical system



Cliques

- A clique is a subset of nodes so that every two distinct nodes in that subset are adjacent.
- A maximal clique is a clique that cannot be expanded by adding more nodes.

Example: Cliques in the dynamical system

- The cliques are the sets $\{1\}$, $\{2\}$, $\{3\}$, $\{4\}$, $\{1,2\}$, $\{2,3\}$ and $\{3,4\}$.
- The maximal cliques are the sets $\{1,2\}$, $\{2,3\}$ and $\{3,4\}$.

More about Markov random fields

General Markov random fields

A Markov random field describes a factorization of a pdf over cliques of the graph:

$$p(\mathbf{x}) = \frac{1}{Z} \prod_{c \in \mathcal{C}} \psi_c(\mathbf{x}_c)$$

Is the MRF unique?

- We can have situations where different collections of maximal cliques are possible.
- We can also include potentials over non-maximal cliques.

We can also have several Bayes nets for a given factor graph.

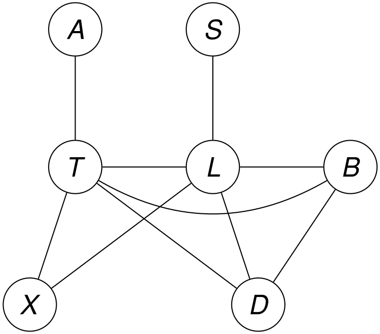
MRFs and belief propagation

Belief propagation for an MRF is similar to belief propagation for a factor graph.

- Instead of products over both factors and variables we only have products over variables.
- An MRF may have cycles which could have been avoided in an equivalent factor graph.

Outline

From Bayes net to Markov random field



To convert a Bayes net to an MRF we must **moralize** all **colliders**.

Colliders (V-structures)

Two nodes that have the same child without being connected is called a collider.

Moralization

Parents of the same child must marry.

The MRF is likely to be more dense than the underlying Bayes net.

For the Chest Clinic network, the **conditional probabilities**

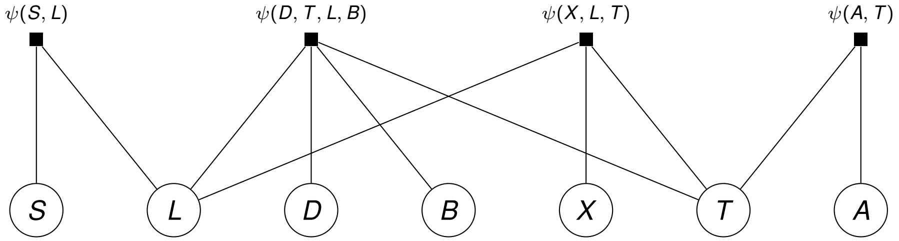
$$p(A) \qquad p(S) \qquad p(B) \qquad p(T|A) \qquad p(L|S) \qquad p(X|T,L) \qquad p(D|T,L,B)$$

are turned into the **potential functions**

$$\psi(T,A) \qquad \psi(L,S) \qquad \psi(X,T,L) \qquad \psi(D,T,L,B).$$

From Bayes net to factor graph

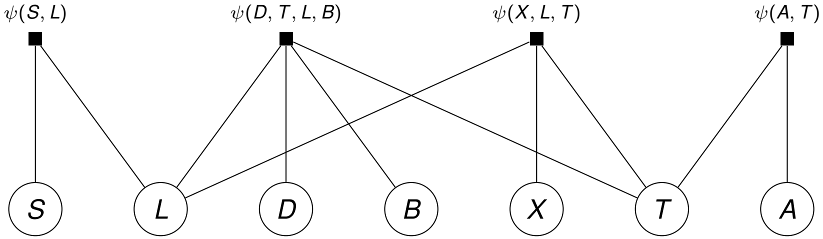
- We can convert all conditional probability functions of the Bayes net to factors.
- . . . or we can make the corresponding MRF first, and convert all its potential functions to factors.



Factor graph of the chest clinic example displayed to emphasize its **bipartite** naturer.

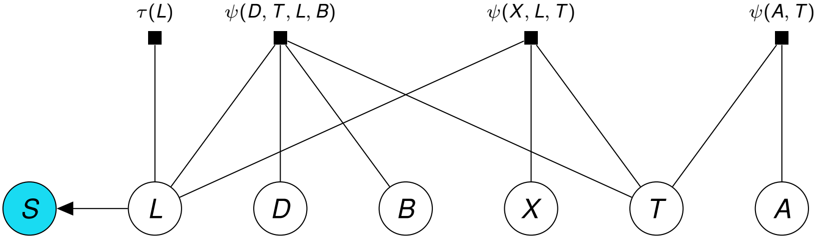
From factor graph to Bayes net (The variable elimination algorithm)

```
for each node $\mathbf{x}_i$ do
    $\mathcal{S}(i) \leftarrow$ all nodes involved in factors adjacent to $i$ except $i$;
    $\phi(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \prod_{c \in \mathrm{ne}(i)} \psi_c(\mathbf{x}_c)$ ;
    $q(\mathbf{x}_i | \mathbf{x}_{\mathcal{S}(i)}) \tau(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \phi(\mathbf{x}_{\mathcal{S}(i)})$ ;
    Replace factors in $\mathrm{ne}(i)$ with $\tau(\mathbf{x}_{\mathcal{S}(i)})$ ;
    Insert $q(\mathbf{x}_i | \mathbf{x}_{\mathcal{S}(i)})$ in Bayes net ;
end
```



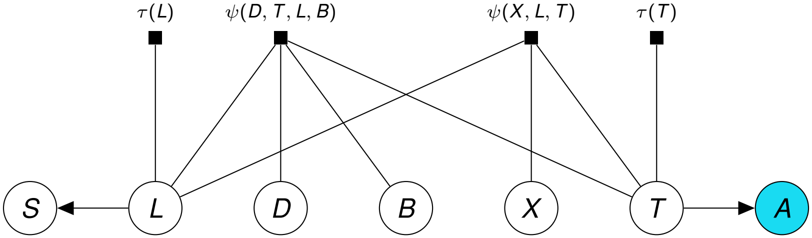
From factor graph to Bayes net (The variable elimination algorithm)

```
for each node $\mathbf{x}_i$ do
    $\mathcal{S}(i) \leftarrow$ all nodes involved in factors adjacent to $i$ except $i$;
    $\phi(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \prod_{c \in \mathrm{ne}(i)} \psi_c(\mathbf{x}_c)$ ;
    $q(\mathbf{x}_i | \mathbf{x}_{\mathcal{S}(i)}) \tau(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \phi(\mathbf{x}_{\mathcal{S}(i)})$ ;
    Replace factors in $\mathrm{ne}(i)$ with $\tau(\mathbf{x}_{\mathcal{S}(i)})$ ;
    Insert $q(\mathbf{x}_i | \mathbf{x}_{\mathcal{S}(i)})$ in Bayes net ;
end
```



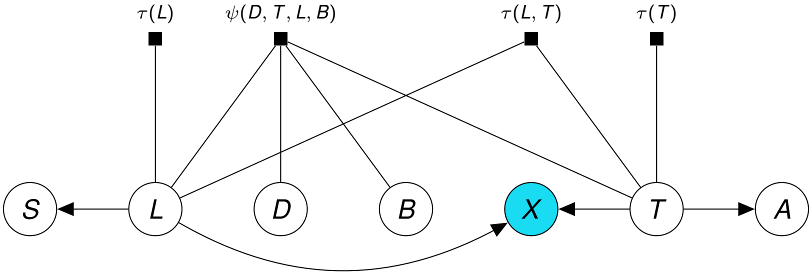
From factor graph to Bayes net (The variable elimination algorithm)

```
for each node $\mathbf{x}_i$ do
    $\mathcal{S}(i) \leftarrow$ all nodes involved in factors adjacent to $i$ except $i$;
    $\phi(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \prod_{c \in \mathrm{ne}(i)} \psi_c(\mathbf{x}_c)$ ;
    $q(\mathbf{x}_i|\mathbf{x}_{\mathcal{S}(i)})\tau(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \phi(\mathbf{x}_{\mathcal{S}(i)})$ ;
    Replace factors in $\mathrm{ne}(i)$ with $\tau(\mathbf{x}_{\mathcal{S}(i)})$ ;
    Insert $q(\mathbf{x}_i|\mathbf{x}_{\mathcal{S}(i)})$ in Bayes net ;
end
```



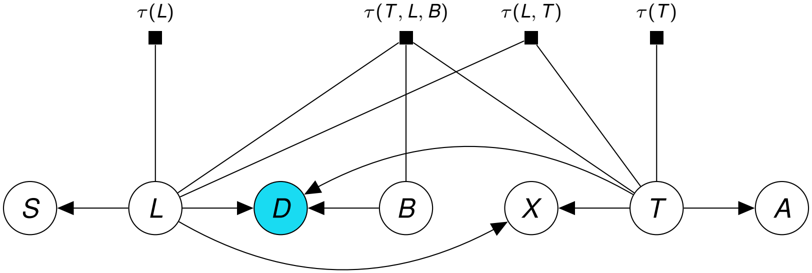
From factor graph to Bayes net (The variable elimination algorithm)

```
for each node $\mathbf{x}_i$ do
    $\mathcal{S}(i) \leftarrow$ all nodes involved in factors adjacent to $i$ except $i$;
    $\phi(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \prod_{c \in \mathrm{ne}(i)} \psi_c(\mathbf{x}_c)$ ;
    $q(\mathbf{x}_i|\mathbf{x}_{\mathcal{S}(i)})\tau(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \phi(\mathbf{x}_{\mathcal{S}(i)})$ ;
    Replace factors in $\mathrm{ne}(i)$ with $\tau(\mathbf{x}_{\mathcal{S}(i)})$ ;
    Insert $q(\mathbf{x}_i|\mathbf{x}_{\mathcal{S}(i)})$ in Bayes net ;
end
```



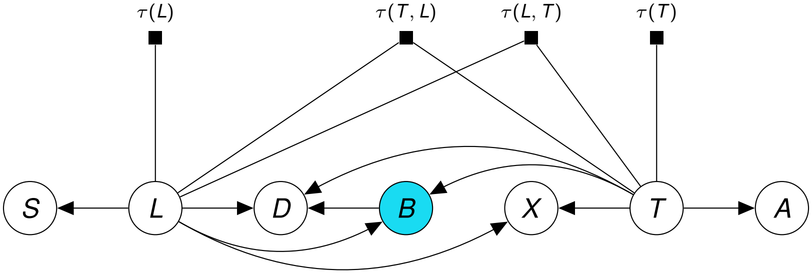
From factor graph to Bayes net (The variable elimination algorithm)

```
for each node $\mathbf{x}_i$ do
    $\mathcal{S}(i) \leftarrow$ all nodes involved in factors adjacent to $i$ except $i$;
    $\phi(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \prod_{c \in \mathrm{ne}(i)} \psi_c(\mathbf{x}_c)$ ;
    $q(\mathbf{x}_i|\mathbf{x}_{\mathcal{S}(i)})\tau(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \phi(\mathbf{x}_{\mathcal{S}(i)})$ ;
    Replace factors in $\mathrm{ne}(i)$ with $\tau(\mathbf{x}_{\mathcal{S}(i)})$ ;
    Insert $q(\mathbf{x}_i|\mathbf{x}_{\mathcal{S}(i)})$ in Bayes net ;
end
```



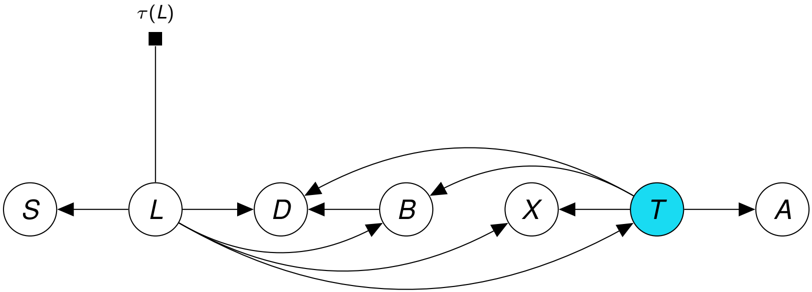
From factor graph to Bayes net (The variable elimination algorithm)

```
for each node $\mathbf{x}_i$ do
    $\mathcal{S}(i) \leftarrow$ all nodes involved in factors adjacent to $i$ except $i$;
    $\phi(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \prod_{c \in \mathrm{ne}(i)} \psi_c(\mathbf{x}_c)$ ;
    $q(\mathbf{x}_i | \mathbf{x}_{\mathcal{S}(i)}) \tau(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \phi(\mathbf{x}_{\mathcal{S}(i)})$ ;
    Replace factors in $\mathrm{ne}(i)$ with $\tau(\mathbf{x}_{\mathcal{S}(i)})$ ;
    Insert $q(\mathbf{x}_i | \mathbf{x}_{\mathcal{S}(i)})$ in Bayes net ;
end
```



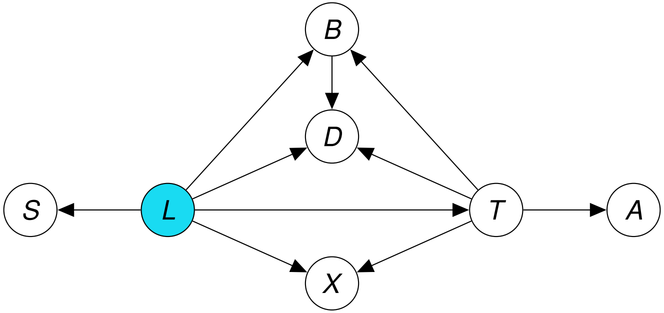
From factor graph to Bayes net (The variable elimination algorithm)

```
for each node $\mathbf{x}_i$ do
    $\mathcal{S}(i) \leftarrow$ all nodes involved in factors adjacent to $i$ except $i$;
    $\phi(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \prod_{c \in \mathrm{ne}(i)} \psi_c(\mathbf{x}_c)$ ;
    $q(\mathbf{x}_i|\mathbf{x}_{\mathcal{S}(i)})\tau(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \phi(\mathbf{x}_{\mathcal{S}(i)})$ ;
    Replace factors in $\mathrm{ne}(i)$ with $\tau(\mathbf{x}_{\mathcal{S}(i)})$ ;
    Insert $q(\mathbf{x}_i|\mathbf{x}_{\mathcal{S}(i)})$ in Bayes net ;
end
```



From factor graph to Bayes net (The variable elimination algorithm)

```
for each node $\mathbf{x}_i$ do
    $\mathcal{S}(i) \leftarrow$ all nodes involved in factors adjacent to $i$ except $i$;
    $\phi(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \prod_{c \in \mathrm{ne}(i)} \psi_c(\mathbf{x}_c)$ ;
    $q(\mathbf{x}_i|\mathbf{x}_{\mathcal{S}(i)})\tau(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \phi(\mathbf{x}_{\mathcal{S}(i)})$ ;
    Replace factors in $\mathrm{ne}(i)$ with $\tau(\mathbf{x}_{\mathcal{S}(i)})$ ;
    Insert $q(\mathbf{x}_i|\mathbf{x}_{\mathcal{S}(i)})$ in Bayes net ;
end
```



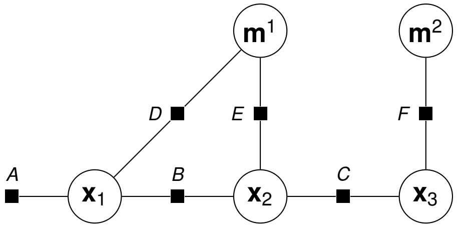
Outline

Graphical models with loops

For graphical models with loops, belief propagation is not guaranteed to give the correct marginals. What can we do?

Alternatives

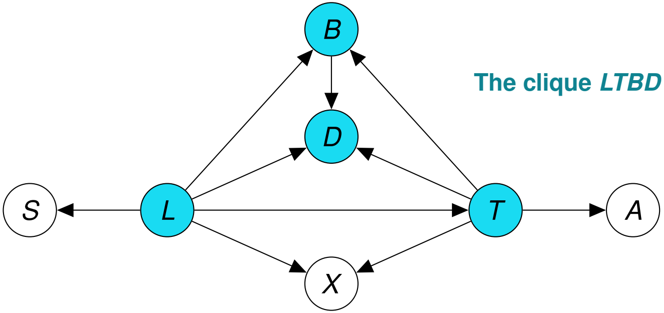
- Brute force evaluation of the joint distribution.
- Use **loopy belief propagation** and hope that it converges to something sensible.
- Generate a **junction tree** whose nodes are maximal cliques of the converted Bayes net. Perform belief propagation between these cliques.



The factor graph of a SLAM problem with 3 poses and 2 landmarks.

Cliques and junction trees

Consider the converted Chest Clinic Bayes net.

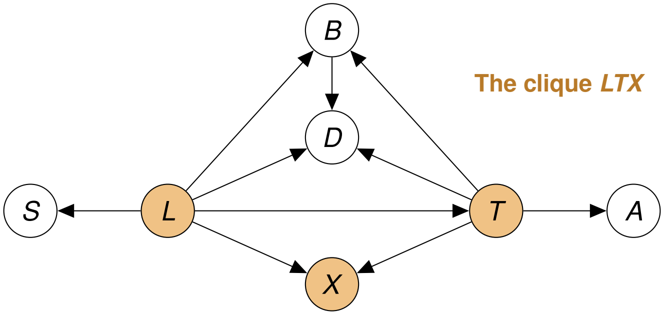


The concept of a junction tree

- A junction tree has two kinds of nodes: Maximal cliques and separators.
- The separator between two cliques is the intersection of nodes in the two cliques.
- It must obey the **running intersection property**: If variable $\mathbf{x}_i$ is in both clique node C_a and clique node C_b, it is also in all nodes on the path between C_a and C_b.

Cliques and junction trees

Consider the converted Chest Clinic Bayes net.

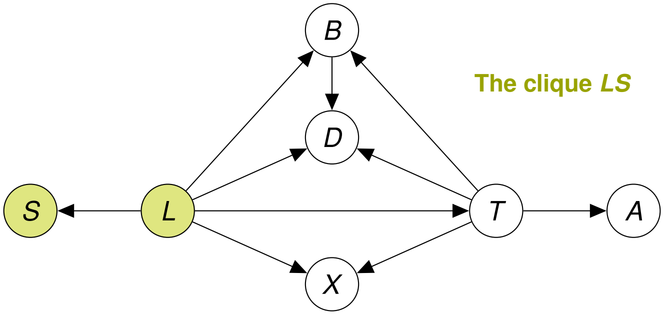


The concept of a junction tree

- A junction tree has two kinds of nodes: Maximal cliques and separators.
- The separator between two cliques is the intersection of nodes in the two cliques.
- It must obey the **running intersection property**: If variable $\mathbf{x}_i$ is in both clique node C_a and clique node C_b, it is also in all nodes on the path between C_a and C_b.

Cliques and junction trees

Consider the converted Chest Clinic Bayes net.



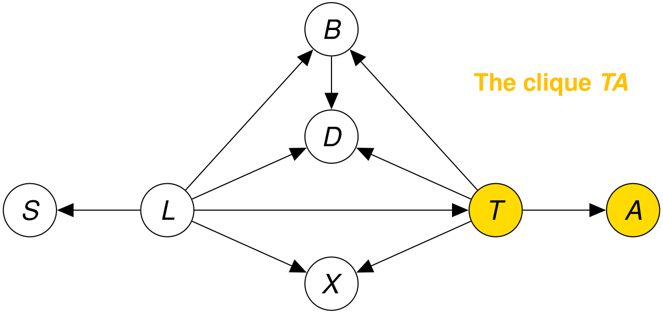
The clique *LS*

The concept of a junction tree

- A junction tree has two kinds of nodes: Maximal cliques and separators.
- The separator between two cliques is the intersection of nodes in the two cliques.
- It must obey the **running intersection property**: If variable $\mathbf{x}_i$ is in both clique node C_a and clique node C_b, it is also in all nodes on the path between C_a and C_b.

Cliques and junction trees

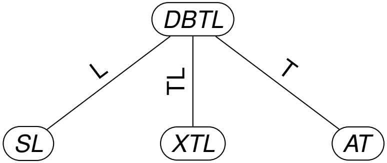
Consider the converted Chest Clinic Bayes net.



The concept of a junction tree

- A junction tree has two kinds of nodes: Maximal cliques and separators.
- The separator between two cliques is the intersection of nodes in the two cliques.
- It must obey the **running intersection property**: If variable $\mathbf{x}_i$ is in both clique node C_a and clique node C_b, it is also in all nodes on the path between C_a and C_b.

Junction trees



Junction tree of the chest clinic example. Separators have been identfied with the edges.

Making a junction tree

To ensure the running intersection property, the clique tree must be a maximum spanning tree: **The sum of cardinalities of the separators is maximized.**

Suggested recipe:

- Convert the factor graph to a Bayes net (it is automatically moral and chordal).
- There will be a unique root node in the Bayes net.
- Find a maximal clique containing the root node.
- Find neighboring maximal cliques and their separators.
- Repeat this process until all nodes and edges of the Bayes net have been covered.

Alternative formulation: The Bayes tree

$$\begin{tikzcd}
C_1 & DBTL \arrow[dl] \arrow[d] \arrow[dr] & \\
S : L & X : TL & A : T \\
C_4 & C_2 & C_3
\end{tikzcd}$$

Bayes tree of the chest clinic example.

The Bayes tree is constructed in the same way as the junction tree.

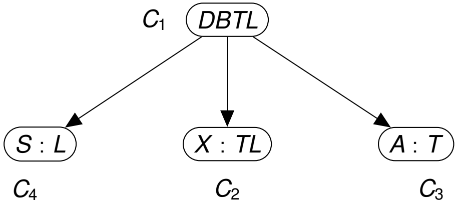
Each clique C_k consist of its **frontal variables** F_k and **separators** S_k. The joint distribution can then be written as

$$p(\boldsymbol{\theta}) = \prod_k p(F_k \mid S_k)$$

Contents of the Bayes tree for the chest clinic example:

Clique no.	F_k	S_k	$p(F_k \mid S_k)$
1	$DBTL$	$\{\}$	$q(D\|TLB)q(B\|TL)q(T\|L)q(L)$
2	X	TL	$q(X \mid LT)$
3	A	T	$q(A \mid T)$
4	S	L	$q(S \mid L)$

Alternative formulation: The Bayes tree


$$\begin{tikzcd} & DBTL \arrow[dl] \arrow[d] \arrow[dr] & \\ S:L & X:TL & A:T \end{tikzcd}$$

Bayes tree of the chest clinic example.

Summary of the Bayes tree algorithm (a.k.a. belief updating)

- Collect-to-root phase: The inward “message passing” towards the root clique is taken care of by the Bayes net conversion.
- Distribute-from-root phase: The outward “message passing” towards the leafs is then simply an exercise in marginalization over nodes in the parent cliques.

Example

$$p(XLT) = \sum_B \sum_D q(X \mid TL)p(DBTL) = q(X \mid TL)q(T|L)q(L)$$

Smoothing revisited: Belief updating for the dynamical system

The forward pass (with the elimination order $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{n-1}, \mathbf{x}_n$)

- We want to factorize the product of current factors and output from last cycle:

$$p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{z}_k \mid \mathbf{x}_k)\tau(\mathbf{x}_{k-1}) \longrightarrow q(\mathbf{x}_{k-1}|\mathbf{x}_k)\tau(\mathbf{x}_k).$$

- It follows that $\tau(\mathbf{x}_{k-1}) = p(\mathbf{x}_{k-1} \mid \mathbf{z}_{1:k-1})$ and furthermore that

$$q(\mathbf{x}_{k-1}|\mathbf{x}_k) = p(\mathbf{x}_{k-1} \mid \mathbf{x}_k, \mathbf{z}_{1:k-1}) \qquad \tau(\mathbf{x}_k) \propto p(\mathbf{x}_k \mid \mathbf{z}_{1:k}).$$

The backward pass

- At the root clique $\{\mathbf{x}_{n-1}, \mathbf{x}_n\}$ we get the calibrated potential

$$\hat{\psi}(\mathbf{x}_{n-1}, \mathbf{x}_n) = p(\mathbf{x}_{n-1} \mid \mathbf{x}_n, \mathbf{z}_{1:n-1})p(\mathbf{x}_n \mid \mathbf{z}_{1:n}) \xrightarrow[\text{Marginalization over } \mathbf{x}_{n-1}]{} p(\mathbf{x}_n \mid \mathbf{z}_{1:n}).$$

- The distribute-from-root phase boils down to marginalization in the converted Bayes net:

$$p(\mathbf{x}_k \mid \mathbf{z}_{1:n}) = \int q(\mathbf{x}_k \mid \mathbf{x}_{k+1})p(\mathbf{x}_{k+1} \mid \mathbf{z}_{1:n})\, \mathrm{d}\mathbf{x}_{k+1}.$$

The Rauch-Tung-Striebel smoother

Assumptions

Assume Gaussian-linear models and that the following information is available:

- The forward filter density $p(\mathbf{x}_k \,|\, \mathbf{z}_{1:k}) = \mathcal{N}(\mathbf{x}_k \,;\, \hat{\mathbf{x}}_k, \mathbf{P}_k)$ from the current time step,
- The backward smoothing density $p(\mathbf{x}_{k+1} \,|\, \mathbf{z}_{1:n}) = \mathcal{N}(\mathbf{x}_{k+1} \,;\, \hat{\mathbf{x}}^s_{k+1}, \mathbf{P}^s_{k+1})$ from the next time step.

The smoother

The current smoothed density $p(\mathbf{x}_k \,|\, \mathbf{z}_{1:n})$ is then given by $\mathcal{N}(\mathbf{x}_k \,|\, \hat{\mathbf{x}}^s_k, \mathbf{P}^s_k)$ where

$$\begin{aligned} \hat{\mathbf{x}}^s_k &= \hat{\mathbf{x}}_k + \mathbf{W}_{k|k+1}(\hat{\mathbf{x}}^s_{k+1} - \mathbf{F}\hat{\mathbf{x}}_k) & \mathbf{W}_{k|k+1} &= \mathbf{P}_k \mathbf{F}^{\mathsf{T}} (\mathbf{P}^-_{k+1})^{-1} \\ \mathbf{P}^s_k &= \mathbf{P}_k + \mathbf{W}_{k|k+1}(\mathbf{P}^s_{k+1} - \mathbf{P}^-_{k+1})\mathbf{W}^{\mathsf{T}}_{k|k+1} & \mathbf{P}^-_{k+1} &= \mathbf{F}\mathbf{P}_k\mathbf{F}^{\mathsf{T}} + \mathbf{Q}. \end{aligned}$$

Why RTS?

In contrast to the two-filter smoother, the RTS smoother uses

- a moment-based parametrization.
- the posterior state estimates and their covariances.

Outline

Front-end and back-end

The front-end

Everything that must be done with the data before they are included in the pose graph.

- Making factors.
- Feature extraction.
- Data association.
- Pre-integration of IMU data.

The back-end

Finds an estimate of map and trajectory by pose graph optimization.

- Typically, the pose graph optimization boils down to finding the MAP estimate of the posterior density $p(\mathbf{x}_{0:k}, \mathbf{m} \,|\, \mathbf{z}_{1:k})$ expressed in terms of a factor graph.
- If $p(\mathbf{x}_{0:k}, \mathbf{m} \,|\, \mathbf{z}_{1:k})$ is close to Gaussian then we expect the joint MAP estimate to coincide with the collection of marginal MAP estimates.
  This is not something that holds for arbitrary distributions!
- To achieve real-time performance, the back-end may exploit relationships between Gauss-Newton optimization, belief updating and sparse matrix algebra.

Cost function of the full SLAM problem

Recall that the "state vector" $\boldsymbol{\eta}$ that we want to estimate for the full SLAM problem consist of the vehicle trajectory $\mathbf{x}_{0:L}$ and the map $\mathbf{m}$.

The posterior of the full range-bearing SLAM problem is

$$p(\mathbf{x}_{0:L}, \mathbf{m} \mid \mathbf{z}_{1:L}) \propto \left(\prod_{k=1}^{L} p(\mathbf{z}_k \mid \mathbf{x}_k, \mathbf{m}) p(\mathbf{x}_k \mid \mathbf{x}_{k-1}) \right) p(\mathbf{x}_0, \mathbf{m})$$

and its logarithm (up to proportionality) can be written as

$$\begin{aligned} s(\mathbf{x}_{0:L}, \mathbf{m}) = & \|\boldsymbol{\eta}_0 - \hat{\boldsymbol{\eta}}_0\|^2_{\mathbf{P}_0} \\ & + \sum_{k=1}^{L} \|\mathbf{f}^{-1}(\mathbf{x}_k, \mathbf{x}_{k-1}) - \mathbf{u}_k\|^2_{\mathbf{Q}} \\ & + \sum_{k=1}^{L} \sum_{i=1}^{m} \|\mathbf{h}(\mathbf{x}_k, \mathbf{m}^i) - \mathbf{z}^i_k\|^2_{\mathbf{R}}. \end{aligned}$$

The goal is to minimize the cost function $s(\mathbf{x}_{0:L}, \mathbf{m})$.

We have used the Mahalanobis distance: $\|\mathbf{x} - \boldsymbol{\mu}\|_{\mathbf{P}} = \sqrt{(\mathbf{x} - \boldsymbol{\mu})^{\mathsf{T}} \mathbf{P}^{-1} (\mathbf{x} - \boldsymbol{\mu})}$.

Factor graph interpretation

Prior factors

The term $\|\boldsymbol{\eta}_0 - \hat{\boldsymbol{\eta}}_0\|^2_{\mathbf{P}_0}$ can hopefully be decomposed into $m + 1$ independent Gaussians.

Odometric factors

We have L factors of the form $\exp(-\frac{1}{2}\|\mathbf{f}^{-1}(\mathbf{x}_k, \mathbf{x}_{k-1}) - \mathbf{u}_k\|^2_{\mathbf{Q}})$.

Landmark factors

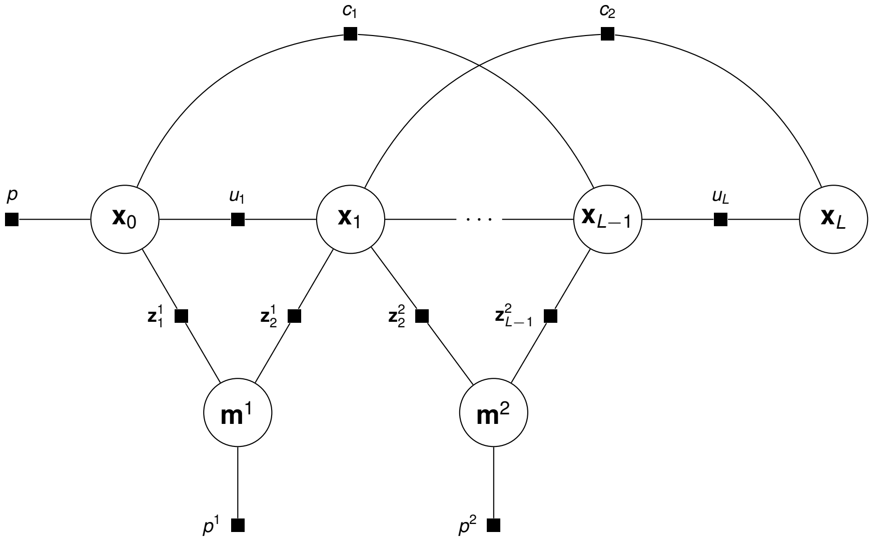
We have up to mL factors of the form $\exp(-\frac{1}{2}\|\mathbf{h}(\mathbf{x}_k, \mathbf{m}^i) - \mathbf{z}^i_k\|^2_{\mathbf{R}})$.

- The number of such factors is likely to be much smaller because not all landmarks are visible from all poses.

$\Rightarrow$ We get more sparsity in the factor graph.

- Notice that there are no landmark-to-landmark factors.
- It is also possible to include special factors for special kinds of measurements (e.g. bearing-only measurements or scan-matching constraints).
- In large scale SLAM systems we may separate landmark factors into local constraints and loop-closure constraints.

Structure of the factor graph



Based on Kaess et al (2012): “ISAM2: Incremental smoothing and mapping using the Bayes tree”, IJRR

The residual vector and the Gauss-Newton update

The key entities involved are the joint state vector $\boldsymbol{\theta}$, the residual vector **e** and the corresponding **information matrix** $\boldsymbol{\Omega}$:

$$\boldsymbol{\theta} = \begin{bmatrix} \mathbf{x}_0 \\ \vdots \\ \mathbf{x}_L \\ \mathbf{m} \end{bmatrix} \qquad \mathbf{e} = \begin{bmatrix} \mathbf{x}_0 - \hat{\mathbf{x}}_0 \\ \mathbf{m} - \hat{\mathbf{m}}_0 \\ \mathbf{f}^{-1}(\mathbf{x}_1, \mathbf{x}_0) - \mathbf{u}_1 \\ \vdots \\ \mathbf{f}^{-1}(\mathbf{x}_L, \mathbf{x}_{L-1}) - \mathbf{u}_L \\ \mathbf{h}(\mathbf{x}_1, \mathbf{m}^1) - \mathbf{z}_1^1 \\ \vdots \\ \mathbf{h}(\mathbf{x}_L, \mathbf{m}^m) - \mathbf{z}_L^m \end{bmatrix} \qquad \boldsymbol{\Omega} = \begin{bmatrix} \mathbf{P}_0^{-1} & & \\ & \mathbf{I}_L \otimes \mathbf{Q}^{-1} & \\ & & \mathbf{I}_{Lm} \otimes \mathbf{R}^{-1} \end{bmatrix}$$

A Gauss-Newton update[1] of the form $\hat{\boldsymbol{\theta}} = \boldsymbol{\theta}^* + \Delta\hat{\boldsymbol{\theta}}$ can then be expressed as

$$\Delta\hat{\boldsymbol{\theta}} = (\mathbf{J}^\top \boldsymbol{\Omega} \mathbf{J})^{-1} \mathbf{J}^\top \boldsymbol{\Omega} \mathbf{e} \qquad \text{where} \qquad \mathbf{J} = \frac{\partial \mathbf{e}}{\partial \boldsymbol{\theta}}$$

is the Jacobian of **e** with respect to $\boldsymbol{\theta}$.

[1] In practice, Gauss-Newton is often replaced with Levenberg-Marquardt, at least in the initial search steps, so that non-convexity can be handled.

The square root information matrix

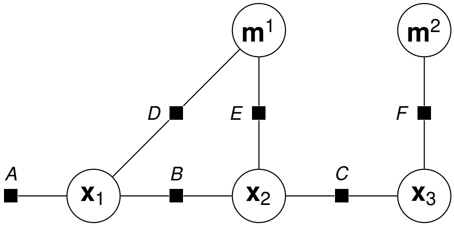
Using the matrix square root ($\boldsymbol{\Omega} = \boldsymbol{\Omega}^{\frac{1}{2}}\boldsymbol{\Omega}^{\mathsf{T}/2}$) it is possible to combine the matrices $\boldsymbol{\Omega}$ and **J** into a single matrix **A** that contains the same information:

$$\mathbf{A} = \boldsymbol{\Omega}^{\mathsf{T}/2}\mathbf{J}$$
$$\mathbf{b} = \boldsymbol{\Omega}^{\mathsf{T}/2}\mathbf{e}.$$

The Gauss-Newton update can then be written as

$$\Delta\hat{\theta} = (\mathbf{J}^{\mathsf{T}}\boldsymbol{\Omega}\mathbf{J})^{-1}\mathbf{J}^{\mathsf{T}}\boldsymbol{\Omega}\mathbf{e} = (\mathbf{A}^{\mathsf{T}}\mathbf{A})^{-1}\mathbf{A}^{\mathsf{T}}\mathbf{b}.$$

The benefit of this is that every factor in the factor graph can be identified with corresponding blocks in A and b.



$$\mathbf{A} = \begin{array}{c|ccccc|c} & \mathbf{m}^1 & \mathbf{m}^2 & \mathbf{x}_1 & \mathbf{x}_2 & \mathbf{x}_3 & \\ \hline & \text{X} & & \text{X} & & & D \\ & \text{X} & & & \text{X} & & E \\ & & \text{X} & & & \text{X} & F \\ & & & \text{X} & & & A \\ & & & \text{X} & \text{X} & & B \\ & & & & \text{X} & \text{X} & C \end{array}$$

The linearized posterior in terms of **A**

Linearization of the residual vector:

$$\mathbf{e}(\boldsymbol{\theta}) \approx \mathbf{e}(\boldsymbol{\theta}^*) + \mathbf{J}\Delta\boldsymbol{\theta}$$

The actual density of $\boldsymbol{\theta}$, which is,

$$p(\boldsymbol{\theta}) = \mathcal{N}(\mathbf{0}\,;\,\mathbf{e}(\boldsymbol{\theta}), \boldsymbol{\Omega}^{-1})$$

is then approximated by

$$\begin{aligned}
p(\Delta\boldsymbol{\theta}) =& \mathcal{N}(\mathbf{J}\Delta\boldsymbol{\theta}\,;\,\mathbf{e}(\boldsymbol{\theta}^*), \boldsymbol{\Omega}^{-1}) \\
\propto& \exp\left(-\frac{1}{2}\|\mathbf{J}\Delta\boldsymbol{\theta} - \mathbf{e}(\boldsymbol{\theta}^*)\|_{\boldsymbol{\Omega}^{-1}}\right) \\
=& \exp\left(-\frac{1}{2}(\mathbf{J}\Delta\boldsymbol{\theta} - \mathbf{e}(\boldsymbol{\theta}^*))^{\mathsf{T}}\boldsymbol{\Omega}^{\frac{1}{2}}\boldsymbol{\Omega}^{\mathsf{T}/2}(\mathbf{J}\Delta\boldsymbol{\theta} - \mathbf{e}(\boldsymbol{\theta}^*))\right) \\
=& \exp\left(-\frac{1}{2}\|\boldsymbol{\Omega}^{\mathsf{T}/2}\mathbf{J}\Delta\boldsymbol{\theta} - \boldsymbol{\Omega}^{\mathsf{T}/2}\mathbf{e}(\boldsymbol{\theta}^*)\|^2\right) \\
=& \exp\left(-\frac{1}{2}\|\mathbf{A}\Delta\theta - \mathbf{b}\|^2\right)
\end{aligned}$$

The QR decomposition

We have found a matrix equivalent of the factor graph. Can we also find a matrix equivalent of the converted Bayes net?

The QR decomposition

Any $c \times n$ matrix $\mathbf{A}$ with $c \geq n$ can be decomposed as

$$\mathbf{A} = \boldsymbol{Q} \begin{bmatrix} \boldsymbol{R} \\ \mathbf{0} \end{bmatrix}$$

where $\boldsymbol{Q}$ is a $c \times c$ orthogonal matrix, and the $n \times n$ matrix $\boldsymbol{R}$ is **upper triangular**.

Inserting this in our least squares problem, and defining $\begin{bmatrix} \mathbf{d} \\ \epsilon \end{bmatrix} = \boldsymbol{Q}^{\mathsf{T}} \mathbf{b}$, yields

$$\|\mathbf{A}\Delta\hat{\theta} - \mathbf{b}\|^2 = \left\| \boldsymbol{Q} \begin{bmatrix} \boldsymbol{R} \\ \mathbf{0} \end{bmatrix} \Delta\hat{\theta} - \mathbf{b} \right\|^2 = \left\| \boldsymbol{Q}^{\mathsf{T}} \boldsymbol{Q} \begin{bmatrix} \boldsymbol{R} \\ \mathbf{0} \end{bmatrix} \Delta\hat{\theta} - \boldsymbol{Q}^{\mathsf{T}} \mathbf{b} \right\|^2 = \|\boldsymbol{R}\Delta\hat{\theta} - \mathbf{d}\|^2 + \|\epsilon\|^2.$$

It follows that the least squares solution to the SLAM problem is equivalent to solving

$$\boldsymbol{R}\Delta\hat{\theta} = \mathbf{d}.$$

The QR factorization: Why and How?

We don't want to calculate $(\mathbf{A}^\mathsf{T}\mathbf{A})^{-1}$ because of its size and denseness.

From $\boldsymbol{R}$ to $\Delta\hat{\theta}$

The upper triangular structure of $\boldsymbol{R}$ means that $\Delta\hat{\theta}$ can be recovered fast from $\boldsymbol{R}\Delta\hat{\theta} = \mathbf{d}$ by means of back-substitution.

Equivalent to the calculation of marginals from the converted Bayes net.

From $\mathbf{A}$ to $\boldsymbol{R}$: 3 equivalent approaches

Because $\boldsymbol{Q}$ is orthogonal, we can express it as a sequence of simpler rotations.

1. Gram-Schmidt orthogonalization: Project every column of $\mathbf{A}$ onto the space of columns so far processed.
2. Givens rotations: Perform rotations involving row i and column j so that

$$\begin{bmatrix} c & -s \\ s & c \end{bmatrix} \begin{bmatrix} a_{ii} \\ a_{ji} \end{bmatrix} = \begin{bmatrix} r_{ii} \\ 0 \end{bmatrix}, \quad \text{and} \quad r_{ii} > 0.$$

3. Householder reflections: At iteration k, reflect remaining columns of $\mathbf{A}[k:c, k:n]$ around bisection between $\mathbf{A}[k:c, k]$ and unit vector $[1, 0, \ldots, 0]^\mathsf{T}$.

ISAM and ISAM2 at a glance

Incremental smoothing and mapping (ISAM)

- QR-based solution of the full SLAM problem purely based on linear algebra.

Kaess, Ranganathan and Dellaert (2008): “iSAM: Incremental Smoothing and Mapping”, IEEE Transactions on Robotics.

ISAM 2

- QR-based solution of the full SLAM problem based on relationships with probabilistic graphical models.
- The Bayes tree data structure is used to achieve more efficient information flow: Only update states affected by the measurements, only re-order and re-linearize when needed, and more efficiently.

Kaess, Johannsson, Roberts, Ila, Leonard and Dellaert (2012): “ISAM2: Incremental smoothing and mapping using the Bayes tree”, International Journal of Robotics Research.

- Implemented through the GTSAM factor graph library (check miniSAM for better Python compatibility).
- Other state-of-the-art graph-based SLAM methods include ORB-SLAM (visual feature-based), LSD-SLAM (visual dense), Google Cartographer (lidar dense).

Incremental inference in ISAM

How to turn our solution to the full SLAM problem into a solution to online SLAM?

Every time a new measurement (factor) is received we append a new block-row $\mathbf{w}^\mathsf{T}$ to $\mathbf{A}$, and a corresponding term γ to $\mathbf{b}$:

$$\begin{bmatrix} \mathbf{A} \\ \mathbf{w}^\mathsf{T} \end{bmatrix} \Delta\hat{\boldsymbol{\theta}} = \begin{bmatrix} \mathbf{b} \\ \gamma \end{bmatrix}.$$

Because $\begin{bmatrix} \boldsymbol{Q}^\mathsf{T} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{A} \\ \mathbf{w}^\mathsf{T} \end{bmatrix} = \begin{bmatrix} \boldsymbol{R} \\ \mathbf{w}^\mathsf{T} \end{bmatrix}$ we can implement this by appending $\mathbf{w}^\mathsf{T}$ to $\boldsymbol{R}$ instead:

$$\begin{bmatrix} \boldsymbol{R} \\ \mathbf{w}^\mathsf{T} \end{bmatrix} \Delta\hat{\boldsymbol{\theta}} = \begin{bmatrix} \mathbf{d} \\ \gamma \end{bmatrix}$$

Re-triangulation

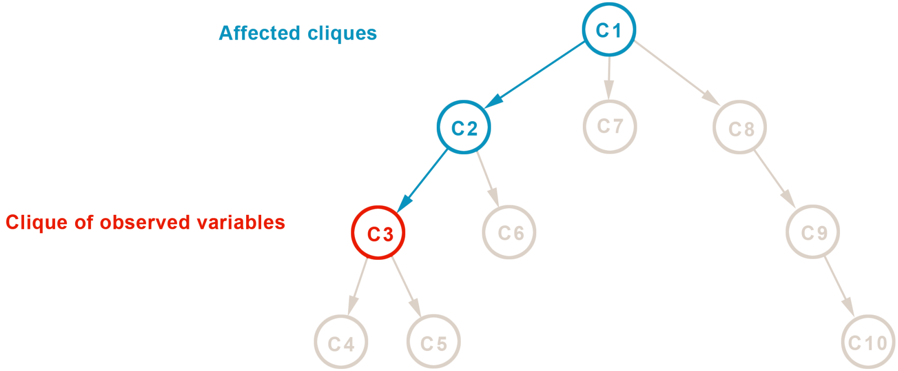
- If $\mathbf{w}^\mathsf{T}$ only involves measurements of existing states, use Givens rotations to eliminate the block from $\boldsymbol{R}$.
- If $\mathbf{w}^\mathsf{T}$ involves a new state variable, perform Givens rotations until only its rightmost element is left.

Incremental inference in ISAM2

- Because the Bayes tree is directed, the densities of the states near the leafs are defined conditional on states closer to the root.

$\Rightarrow$ When we update our representation of the Bayes tree (i.e., $\boldsymbol{R}$) we only have to update states that are between the observed states and the root.

- This has important consequences for **reordering** and **relinearization**.



Reordering

Reordering the columns in A can affect how much fill-in there will be in *R*.

Re-ordering in ISAM

- Use a standard heuristic, such as the Column Approximate Minimum Degree (COLAMD) algorithm, which analyses the matrix **A** to find a good ordering.
- Periodic batch reorderings.

Re-ordering in ISAM2

- Use a constrained version of the COLAMD algorithm.
- The main constraint is that recently accessed variables should come towards the end of the ordering.
- Reordering is done after incremental updates, only for the affected variables.

Relinearization

As we append new rows to *R* and retriangulize we insert information in the system which will make the old linearization less valid.

Relinearization in ISAM

Periodic batch relinearizations.

When we relinearize we have to perform the entire QR-decomposition again.

Relinearization in ISAM2

- “Fluid relinearization”: Relinearize only when the estimate of a variable differs more than a given threshold from its linearization point.
- We only have to reconstruct the QR-decomposition for the branch between the relinearized variables and the root.

Data association in feature-based Graph-SLAM

Non-probabilistic approaches

- Use Random Sample and Consensus (RANSAC).
- Match descriptors of features (e.g., done in ORB-SLAM).

Probabilistic approaches

We can still calculate the innovation covariance used in JCBB, but it is more complicated than in EKF-SLAM.

- The innovation covariance used by JCBB can be constructed as

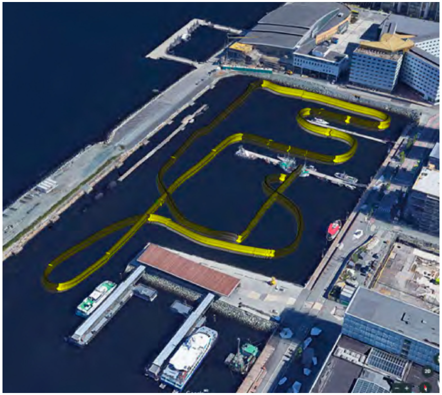
  $$\mathbf{S} = \mathbf{J}_a(\boldsymbol{R}^\mathsf{T}\boldsymbol{R})^{-1}\mathbf{J}_a^\mathsf{T} + \mathbf{R}_a$$

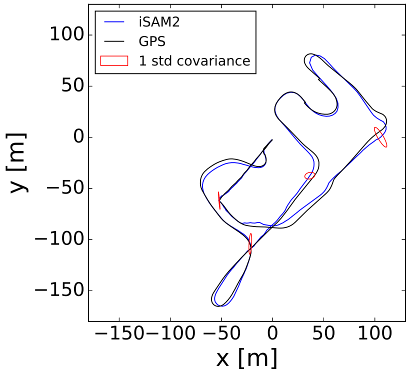
  where $\mathbf{R}_a$ and $\mathbf{J}_a$ are the measurement noise and Jacobian matrices corresponding to the association hypothesis a.
- The cross-covariances between the current pose and the landmarks can be found by solving

  $$\boldsymbol{R}^\mathsf{T}\boldsymbol{R}\,\mathbf{X} = \begin{bmatrix}\mathbf{0}_{n-3\times 3}\\ \mathbf{I}_3\end{bmatrix}.$$
- A more complicated dynamic programming scheme is needed to calculate marginal landmark covariances.

Example of ISAM2 for lidar-based maritime SLAM

- Point features extracted from lidar using the Point Cloud Library (PCL).
- Data association by means of RANSAC and feature descriptors from the PCL.





Skjellaug, Even (2020): “Feature-Based Lidar SLAM for Autonomous Surface Vehicles Operating in Urban Environments “, MSc thesis, NTNU,
`http://folk.ntnu.no/edmundfo/msc2020-2021/Skjellaug_SLAM.pdf`.